OpenH323 Gatekeeper - El GNU Gatekeeper

Este manual es traducido al español por: Luis Fernando Ramirez Cueva. <lufer_ramirez@yahoo.es>
Version 2.2.5, December 2006

Éste Manual de Usuario explica cómo compilar, instalar, configurar y monitorear el GNU Gatekeeper (GnuGk).

Índice

1. Introducción

1.1. Acerca de

OpenH323 Gatekeeper - El GNU Gatekeeper `<http://www.gnugk.org/>` es un proyecto de código abierto que implementa un Gatekeeper H.323. Un gatekeeper ofrece servicios de control de llamada a los clientes H.323. Éste es una parte integral de las instalaciones telefónicas en internet más utilizadas basadas en el estándar H.323.

De acuerdo con la Recomendación H.323, un gatekeeper debe ofrecer los siguientes servicios:

- Traducción de direcciones.
- Control de admisiones.
- Control de ancho de banda.
- Administración de zonas.
- Control de señalizacion de las llamadas.
- Autorizacion de llamadas.
- Administración de ancho de banda.
- Administración de llamadas.

El GNU Gatekeeper implementa la mayor parte de estas funciones basadas en la pila de protocolos *OpenH323* `<http://sourceforge.net/projects/openh323/>`

La Recomendación H.323 es un estándar internacional publicado por el *ITU* `<http://www.itu.int/>`. Es un estándar de comunicaciones para audio, video y datos a traves de Internet. Puedes consultar también *A primer on the H.323 Series Standard* `<http://www.packetizer.com/voip/h323/papers/primer/>` de Paul Jones.

Para una descripcion detallada de lo que hace un gatekeeper consulte *aqui* `<http://www.iec.org/online/tutorials/h323/topic06.html>`.

1.2. Copyright

El GNU Gatekeeper como su nombre lo indica, está publicado bajo la licencia *GNU General Public License* (GNU GPL). Además, otorgamos explicitamente el derecho a enlazar este código a las librerias OpenH323 y OpenSSL.

Generalmente hablando, la GNU GPL le permite copiar, distribuir, revender o modificar este software, pero requiere que los trabajos derivados sean publicados también bajo la GNU GPL. Esto significa que debe publicar el código fuente completo de todas las extensiones hacia el gatekeeper y de todos los programas que incluyan dentro al gatekeeper. Consulte el fichero COPYING para mas detalles.

Si ésto no es lo que usted desea, usted debe conectarse al gatekeeper a través del puerto de estado y comunicarse con éste via TCP. De esta forma usted solo tiene que integrar la funcionalidad básica dentro del gatekeeper (y facilitar el código fuente de ésta) y así puede mantener otras partes de su aplicacion bajo otra licencia.

1.3. Nombre

El nombre formal de este proyecto es *OpenH323 Gatekeeper - El GNU Gatekeeper*, de manera abreviada es *GnuGk*. Por favor no confunda este proyecto con otros proyectos de gatekeeper.

Hay muchos proyectos de gatekeepers open-source basados en la pila del protocol OpenH323.

- *OpenGatekeeper* `<http://opengatekeeper.sourceforge.net/>`

  Un gatekeeper disponible gratuitamente bajo MPL. Este proyecto ha estado inactivo por un tiempo.

- *OpenGK* <http://sourceforge.net/projects/openh323>

  Solo en fase inicial.

- *OpenH323 Gatekeeper - El GNU Gatekeeper* <http://www.gnugk.org/>

  Este mismo, además conocido con el nombre de GnuGk.

Tener diferentes gatekeepers con nombres muy similares es confuso para la mayoría de los usuarios. Debido a que nuestro .ºpenH323 Gatekeeper"fué el primero en aparecer, no es nuestra culpa que otros hayan elegido nombres similares. Pero para que la distinción sea un poco mas clara sin confundir a la gente todavía mas, hemos decidido darle un subtítulo al proyecto .ºpenH323 Gatekeeper - El GNU Gatekeeperz empezar a usar `gnugk` como nombre para los ejecutables.

1.4. Descargas

La versión más estable y la versión en desarrollo están disponibles en *la página de descargas* <http://www.gnugk.org/h323download.html> del GnuGk.

La última versión del código fuente está en el CVS en *Sourceforge Web-GUI* <http://openh323gk.cvs.sourceforge.net/openh323gk/openh323gk/>. Recuerde - Esta versión puede contener muchos fallos aun.

También puede descargar algunos ejecutables desde *la pagina de descargas* <http://www.gnugk.org/h323download.html>.

1.5. Listas de correo

Existen dos listas de correo para el proyecto, una para los desarrolladores y otra para usuarios.

Las preguntas de los usuarios deben de ser enviadas a *lista para usuarios* <mailto:Openh323gk-users@sourceforge.net>. Puede encontrar un archivo de la lista de correo *aqui* <http://sourceforge.net/mailarchive/forum.php?forum_id=8549>. Para unirse a la lista de correo, pulse *aqui* <http://lists.sourceforge.net/lists/listinfo/openh323gk-users>.

Para notificar problemas o enviar bugs/parches, envíe los mails a la *lista para desarrolladores* <mailto:Openh323gk-developer@sourceforge.net>. El archivo de la lista está *aqui* <http://sourceforge.net/mailarchive/forum.php?forum_id=3079>. Por favor envie las preguntas de usuario a la lista para usuarios y deje esta lista solo para desarrolladores!. Si desea contribuir al proyecto, por favor *únase a la lista* <http://lists.sourceforge.net/lists/listinfo/openh323gk-developer>.

Nota: Por favor no envíe preguntas a los mails privados de los desarrolladores. Normalmente estamos muy ocupados. **No** nos gustaria ser su asesor privado, a no ser que desee pagarnos. Envíe sus problemas a la lista de correo pública apropiada para que todo el mundo pueda ayudarle.

Por favor, no envíe tampoco los problemas especificos de GnuGk a la lista de OpenH323, o viceversa. Son proyectos diferentes, aunque muy relacionados.

Antes de enviar un email, asegúrese de que ha leido los documentos relacionados detenidamente. Describa sus problemas de forma clara y precisa. Muestrenos los mensajes de error o registros si los hay.

1.6. Colaboradores

El coordinador del proyecto actual es *Jan Willamowius* <http://www.willamowius.de/> <*jan@willamowius.de*>

Las principales características y funciones de la versión 2.0 lo son gracias a Chih-Wei Huang <*cwhuang@linux.org.tw*> y *Citron Network Inc.* <http://www.citron.com.tw/>, incluyendo el registro a traves de thread-safe y tablas de llamada, nueva arquitectura de modo de enrutado, proxy H.323, H.235 authentication y MySQL backend.

Michal Zygmuntowicz <*m.zygmuntowicz@onet.pl*> ha hecho un gran trabajo en el soporte de Radius y otras mejoras.

La version inicial del gatekeeper fue desarrollada por Xiang Ping Chen, Joe Metzger y Rajat Todi.

2. Compilación e Instalación

2.1. Compilando el Gatekeeper

Para construír el gatekeeper se necesitan al menos PWLib 1.5.0 y OpenH323 1.12.0 o superiores. La version en desarrollo del gatekeeper normalmente necesita la última version disponible de OpenH323. Esas librerias estan disponibles en la *Página de descargas de OpenH323* <http://sourceforge.net/projects/openh323>. Consulte las instrucciones sobre *Cómo compilar el código OpenH323* <http://www.voxgratia.org/docs/faq.html>.

Orden de compilacion:

1. Librería PWLib (versiones release y/o debug)
2. Librería OpenH323
3. Pruebas a la aplicación OpenH323 (no es necesaria, solo para asegurarse de que todo funciona correctamente)
4. El Gatekeeper

En Unix haga un `configure` y `make debug` o `make opt` en el directorio del gatekeeper para crear la version debug o release, respectivamente. Use `make both` para crear ambas versiones. Recuerde que tiene que usar el GCC 3.3.x o superior. Las versiones anteriores pueden no funcionar. Una buena práctica es hacer un `make debugdepend` o `make optdepend` en el directorio del gatekeeper antes de empezar la compilacion actual (make debug o make opt) - estos comandos crean las listas de dependencias apropiadas, asi despues de que actualice las fuentes desde el CVS, todos los ficheros afectados seran recompilados. De lo contrario puede acabar con el Gatekeeper compilado parcialmente con las antiguas cabeceras y parcialmente con las cabeceras actualizadas - algo muy malo.

En Windows simplemente abra y compile la solución facilitada (`gk.sln`) para Microsoft Visual Studio .NET 2003 o el workspace (`gk.dsw`) para Microsoft Visual Studio 6.0 SP6. Por supuesto necesita tener las librerías PWLib y OpenH323 yá compiladas y establecidos los paths include/library apropiados. Si usted desea tener soporte para MYSQL o PostgreSQL instale/compile las librerias del cliente apropiadas y agregue HAS_MYSQL=1 y/o HAS_PGSQL=1 al preprocesor macros del proyecto del gatekeeper. Además necesita indicarle al compilador donde encontrar los include files/libraries y decirle al linker que se enlace con esas librerías del cliente.

Teclee `configure -help` para ver una lista detallada de todas las opciones de compilacion. Puede usarlas para activar o desactivar algunas caracteristicas del gatekeeper. Por ejemplo, si no necesita us RADIUS simplemente teclee: `configure -disable-radius`.

Las versiones recomendadas de las librerías PWLib/OpenH323 son aquellas de la release Pandora (1.7.5.2/1.14.4.2) o superiores. Versiones anteriores ya no son soportadas y no se garantiza que funcionen con el gatekeeper.

Para construír el gatekeeper con las librerias de OpenH323 y del sistema enlazadas estaticamente, tiene que usar `make optnoshared` o `make debugnoshared`.

Para poder usar el gatekeeper en entornos de gran carga, es recomendado habilitar la caracteristica LARGE_FDSET (configure –with-large-fdset=1024) (SOLAMENTE PARA VERSIONES UNIX). Algunos sistemas además necesitan utilizar el ülimit"para permitir que más de 1024 sockets sean asignados a un solo proceso. Tenga en cuenta que la librería PWLib, empezando desde la versión 1.5.3 soporta una característica similar también, de esta manera usted puede seleccionar entre el LARGE_FDSET del GnuGk y la implementación de la librería PWLib. La implementación nativa del GnuGk proporciona mejores resultados de rendimiento. El máximo valor de LARGE_FDSET debe ser calculado basándose en la predicción del máximo uso de sockets. Una regla al ojo puede ser:

```
MAX_NUMBER_OF_CONCURRENT_CALLS * 10 * 120%

Donde:
10 = 2 sockets para Q.931 + 2 sockets para H.245 + 6 sockets para RTP y otras cosas
```

Asi para 100 llamadas simultaneas no necesita mas de 1024 sockets en el LARGE_FDSET.

2.2. La Herramienta .ªddpasswd"

La autenticación del acceso a la línea de estado y el módulo SimplePasswordAuth requieren que contraseñas encriptadas sean almacenadas en el archivo de configuración del gatekeeper. Además desde la versión 2.2.2, el gatekeeper soporta encriptación de todas las contraseñas en la configuración. La utilidad `addpasswd` ayuda a generar y almacenar esas contraseñas encriptadas. Esta utilidad está incluída con el gatekeeper y puede ser compilada empleando:

```
$ make addpasswd
```

El uso de esta utilidad es como se sigue:

```
$ addpasswd CONFIG SECTION KEYNAME PASSWORD
```

Ejemplo 1: El usuario 'gkadmin' con la contraseña 'secret' tiene que ser agregado a la sección de configuración [GkStatus::Auth] para habilitar la autenticación de la interfaz de la linea de estado:

```
$ addpasswd gnugk.ini GkStatus::Auth gkadmin secret
```

Ejemplo 2: El usuario 'joe' con la contraseña 'secret' tiene que ser agregado a la sección de configuración [Password] para habilitar la autenticación de endpoints:

```
$ addpasswd gnugk.ini Password joe secret
```

Ejemplo 3: Una "secret.^{en}criptada compartida es agregada a la sección de configuración RadAuth:

```
$ addpasswd gnugk.ini RadAuth SharedSecret VerySecretPassword
```

IMPORTANTE: La variable `KeyFilled` define un valor por defecto para la clave (key) de encriptación de contraseñas. Esta puede omitirse en la configuración (entonces se define a 0), pero si ésta es especificada, cada vez ésta cambia, las contraseñas encriptadas tienen que ser regeneradas (encriptadas nuevamente utilizando la utilidad `addpasswd`). Además tenga presente que al usar esta utilidad se borrarán todas las líneas que estén comentadas dentro del archivo de configuración.

2.3. Instalación del Gatekeeper

No se necesita un procedimiento especial de instalacion. Simplemente copie el ejecutable en el directorio que usted quiera y cree un fichero de configuración para este. Hay muchos ejemplos de configuracion en el subdirectorio `etc/` del arbol del codigo fuente. Consulte la seccion 4.2 (Archivo de Configuration) para una explicacion mas detallada.

Por ejemplo, en una plataforma Linux x86, el ejecutable optimizado `gnugk` es creado en el subdirectorio `obj_linux_x86_r/`. Puede copiarlo a `/usr/sbin/`, crear una configuracion en `/etc/gnugk.ini` y ejecutarlo mediante

```
$ /usr/sbin/gnugk -c /etc/gnugk.ini -o /var/log/gnugk.log -ttt
```

Consulte la seccion 4.1 (Opciones de línea de comandos) para más detalles.

2.4. Binarios Pre-Construídos.

Si no desea compilar el gatekeeper desde el código fuente, hay muchos 'paquetes' pre-construídos disponibles en *SourceForge* `<http://sourceforge.net/project/showfiles.php?group_id=4797>`. No todas las versiones serán puestas a disposición como binarios, compruebe cual está disponible.

Paquetes Tar (.tgz or .tar.gz)

Descarge el fichero tar e intruduzca el siguiente comando bajo `root`, sustituya el nombre del fichero por el del que ha descargado.

```
$ tar xvzf gnugk-x.x.x.tar
```

Paquetes Debian (.deb)

Debian incluye el gatekeeper como el paquete openh323gk. Puede instalarlo usando el siguiente comando bajo `root`:

```
$ apt-get install openh323gk
```

3. Empezando el Tutorial

3.1. Un simple primer experimento

Para ver que todos los componentes están ejecutándose y funcionando, escoja 2 estaciones de trabajo Linux, ambas conectadas a la LAN. Asegúrese de que tiene al menos la versión 1.1 de OpenH323 y OhPhone instalados. En la primera máquina ejecute el gatekeeper y el ohphone (en consolas diferentes:

```
jan@machine1 > gnugk -ttt
```

Ahora el gatekeeper está ejecutándose en modo directo. La opción `"-ttt"le dice al gatekeeper que genere mucha salida para debug en la consola (puede dirigir esa salida a un fichero con "-o logfile").`

```
jan@machine1 > ohphone -l -a -u jan
```

Ahora el OhPhone está escuchando (-l) para recibir llamadas y las aceptará automáticamente (-a). Se ha registrado como usuario jan con el gatekeeper que será detectado automáticamente. (si la autodetección fallara por alguna razón, use "-g 1.2.3.4"para especificar la IP donde está escuchando el gatekeeper).

En la segunda maquina ejecute solamente ohphone:

```
peter@machine2 > ohphone -u peter jan
```

La segunda instancia de OhPhone se registra con el autodetectado gatekeeper como usuario peter e intenta llamar al usuario jan. El gatekeeper resolverá el nombre de usuario a la IP desde donde el usuario jan se registró (máquina1 en este caso) y OhPhone llamará la otra instancia de OhPhone en la maquina uno.

La primera instancia de OhPhone aceptará esta llamada y Peter y Jan podran hablar.

3.2. Uso del Puerto de Estado para Monitorear el Gatekeeper

Ahora intente ver qué mensajes son manejados por el gatekeeper. En la nueva consola en maquina1 vamos a usar telnet para conectar al gatekeeper:

```
jan@machine1 > telnet machine1 7000
```

Lo más probable es que recibiremos un mensaje de acceso denegado (.Äccess forbidden!"), porque no se le permite a todo el mundo espiar.

Ahora crearemos un fichero llamado gatekeeper.ini y lo colocaremos en el directorio donde arrancamos el gatekeeper. gatekeeper.ini solo contiene 4 lineas:

```
[Gatekeeper::Main]
Fortytwo=42
[GkStatus::Auth]
rule=allow
```

Detenga el gatekeeper con Ctrl-C y ejecútelo de nuevo. Cuando hagamos el telnet de nuevo, estaremos conectados con el gatekeeper. Ahora repita el primer experimento donde Peter llama a Jan y observe qué mensajes son manejados por el gatekeeper en modo no enrutado. Hay un número de comandos que pueden ser emitidos en esta sesión telnet: Teclee "help"para verlos. Para finalizar la sesión de telnet con el gatekeeper teclee "quitz presione Enter.

Pero esto es muy inseguro, todos podrían conectarse hacia la interfaz de estado y ver ésta. Cambiemos entonces el archivo de configuración de la siguiente manera

```
[Gatekeeper::Main]
Fortytwo=42
[GkStatus::Auth]
rule=password
gkadmin=QC7VyAo5jEw=
```

La quinta línea es agregada por la utilidad addpasswd, ésta crea un usuario llamado "gkadminçon una contraseña "secret"limitando de esta manera el acceso hacia el puerto de estado. Reinicie el gatekeeper con esta nueva configuración y realice telnet nuevamente.

Ahora se le preguntará por un nombre de usuario (username) y una contraseña (password) antes de que pueda entrar.

Eche una mirada a la sección 4.2.2 (GkStatus::Auth) para más detalle sobre seguridad del puerto de estado.

3.3. Arrancando el gatekeeper en modo enrutado.

Arrancar o ejecutar el gatekeeper en modo enrutado significa que el gatekeeper utiliza "señales de enrutado gatekeeper"para todas las llamadas. En este modo de gatekeeper todos los mensajes de señalización pasarán a través del gatekeeper y esto permite mucho mas control sobre las llamadas.

```
jan@machine1 > gnugk -r
```

Ahora el gatekeeper está ejecutándose en modo enrutado. Haga telnet al puerto de estado y realice una llamada para ver que mensajes son manejados ahora por el gatekeeper.

Recuerde que todos los paquetes multimedia (audio y video) son todavía enviados directamente entre los endpoints (las 2 instancias de ohphone).

3.4. Un PBX virtual: Desconectando llamadas

Hasta ahora el gatekeeper ha actuado solamente como un mecanismo para resolver nombres simbólicos a direcciones IP. Es una función importante pero difícilmente excitante.

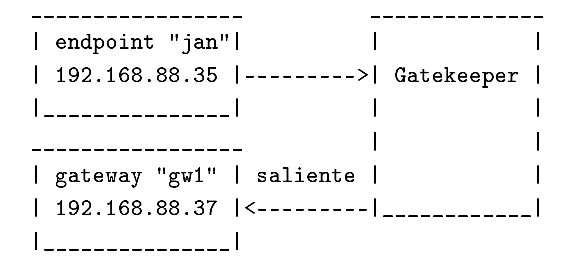
Puesto que el gatekeeper tiene un gran control sobre las llamadas, éste puede finalizarlas por ejemplo. Cuando estamos conectados al Puerto de Estado, podemos listar todas las llamadas activas con "PrintCurrentCalls". Para finalizar una llamada, podemos decir "Disconnectip 1.2.3.4"para uno de estos endpoints.

Uno puede por ejemplo escribir un script simple que conecte al puerto de estado y escuche todas las llamadas entrantes y las termine después de 5 minutos, así ningún usuario puede sobrepasar los recursos del sistema.

Eche un vistazo a las otras funciones de telefonía como TransferCall para ver que más hay disponible.

3.5. Enrutando llamadas hacia un gateway para alcanzar usuarios externos

Sin usar un gateway usted solo puede llamar a otras personas con un teléfono IP a traves de internet. Para alcanzar gente con teléfonos convencionales debe de usar un gateway.



El gatekeeper tiene que conocer qué llamadas deben de ser supuestamente enrutadas a través del gateway y que números deben ser llamados directamente. Use la sección [RasSrv::GWPrefixes] del fichero de configuración para decirle al gatekeeper el prefijo de números que deben de ser enrutados a través del gateway.

```
[RasSrv::GWPrefixes]
gw1=0
```

Esta entrada le dirá al gatekeeper que enrute todas las llamadas con numeros E.164 que empiezen por 0 al gateway que se ha registrado con el alias "gw1". Si no hay ningun gateway registrado con este alias la llamada fallará. Recuerde que debe usar un alias para el gateway - no puede simplemente decirle al gatekeeper la IP del gateway.

Un prefijo puede contener los dígitos 0-9, # y *. Tambien puede contener un caracter especial . (un punto) que sustituye cualquier dígito y puede ser prefijado con ! (un signo de exclamación final) para desactivar el prefijo. La sustitución de prefijos se hace de acuerdo a la regla de sustitución de prefijos mas larga, teniendo las reglas ! mayor prioridad si la longitud es igual. Algunos ejemplos:

```
[RasSrv::GWPrefixes]
; Esta entrada enrutará numeros que empiecen con 0048 (pero no con 004850 y 004860)
; al gw1
gw1=0048,!004850,!004860
; Esta entrada hará corresponder solamente 001 y 10 digitos a continuación
gw2=001..........
```

3.6. Reescribiendo números E.164

Mientras este usando un gateway frecuentemente tendrá que usar internamente numeros diferentes y reescribirlos antes de enviarlos a través del gateway a una red telefónica. Puede usar la sección 6.3 (RasSrv::RewriteE164) para configurar ésto.

Ejemplo: Usted quiere llamar el número 12345 con su telefono IP y le gustaría alcanzar el número 08765 detrás de un gateway llamado "gw1".

```
[RasSrv::GWPrefixes]
gw1=0

[RasSrv::RewriteE164]
12345=08765
```

Usted puede además configurar la reescritura de números E.164 basándose desde qué gateway usted está recibiendo la llamada o enviando una llamada para que utilice la sección 6.4 (RasSrv::GWRewriteE164).

Ejemplo: Usted tiene dos gateways diferentes ("gw1z "gw2") hacia los cuales usted está enviando llamadas con prefijos 0044, pero los mismos requieren que sean agregado un prefijo diferente en el número después de que el ruteo ha elegido el gateway. Esto podría ser para propósitos de identificación por ejemplo.

```
[RasSrv::GWPrefixes]
gw1=0044
gw2=0044
```

```
[RasSrv::GWRewriteE164]
gw1=out=0044=77770044
gw2=out=0044=88880044
```

Ejemplo: Usted necesita identificar llamadas desde un gateway particular "gw1çon un prefijo específico antes que pasen estas llamadas hacia otro gateway "gw2".

```
[RasSrv::GWPrefixes]
gw2=1

[RasSrv::GWRewriteE164]
gw1=in=00=123400
```

La reescritura de expresiones acepta los caracteres comodín como '.' y signos de porcentaje '%' para permitir construír reglas más generales. El caracter punto puede estar en cuanquier lado de las expresiones (izquierda o derecha), el signo de porcentaje solo puede estar a la izquierda. Use '.' para corresponder cualquier caracter y copiar éste en la cadena reescrita y '%' para emparejar cualquier caracter e ignorar éste. Aqui uno ejemplos simples:

```
[RasSrv::RewriteE164]
; Reescribe 0044 + 7 dígitos mínimo hacia 44 + 7 dígitos mínimo
0044.......=44.......
; Reescribe números que empiezan con 11 + 4 dígitos + 11  hacia 22 + 4 dígitos + 22
; (así 11333311 => 22333322, 110000112345 => 220000222345)
11....11=22....22
; quita los primeros cuatro dígitos de todos los nñumeros (11114858345 => 4858345)
; esto es equivalente a 10 reglas %%%%1=1, %%%%2=2, ...
%%%%.=.
; inserta dos ceros en medio del núumero (111148581234 => 11110048581234)
....48=....0048
; incluso esto es posible (415161 => 041051061)
4.5.6=04.05.06
```

4. Configuración Básica del Gatekeeper

El comportamiento del gatekeeper está completamente determinado por las opciones de la línea de comandos y del archivo de configuración. Algunas opciones de la línea de comandos pueden sustituir a opciones del archivo de configuración. Por ejemplo, la opción -l sustituye al atributo TimeToLive del archivo de configuración.

4.1. Opciones de Línea de Comandos

Casi todas las opciones tienen una forma corta y una larga, por ejemplo, -c es lo mismo que -config.

4.1.1. Lo Básico

-h -help

Muestra todas las opciones disponibles y sale del programa.

-c -config filename

Especifica el archivo de configuración a usar.

-s -section section

Especifica cual sección del archivo de configuración será utilizada como principal. Por defecto es [Gatekeeper::Main].

-i -interface IP

Especifica la interfaz (número IP) por el que el gatekeeper escuchará. Usted puede omitir esta opción para permitir que el gatekeeper determine automáticamente la IP en la que escuchará, a menos que desees que el gatekeeper solo se enlace a una IP específica.

-l -timetolive n

Especifica el tiempo de vida (en segundos) para el registro de los endpoints. Este prevalece sobre el atributo TimeToLive del archivo de configuración. Revisar 4.2.1 (aquí) para una explicación detallada.

-b -bandwidth n

Especifica el total de ancho de banda disponible para el gatekeeper. Si no se especifica esta opción, la administración de ancho de banda se desactiva por defecto.

-pid filename

Especifica el archivo pid, solo válido para versiones Unix.

-u -user name

Ejecuta el gatekeeper como dicho usuario. Solo válido para versiones Unix.

-core n

(Unix solamente) Habilita la escritura en archivos çore dumpçuando la aplicación finaliza de forma incorrecta. Un archivo core dump no sobrepasará el tamaño de n bytes. Una constante especial ünlimited"puede ser utilizada para no forzar un límite particular.

4.1.2. Modos del Gatekeeper

Las opciones en esta subsección sustituyen la configuración en la sección 5.1 ([RoutedMode]) del archivo de configuración.

-d -direct

Utiliza señalización de llamadas directamente entre endpoints.

-r -routed

Utiliza señalización de llamadas enrutadas a través del gatekeeper.

-rr -h245routed

Utiliza señalización de llamadas y canal de control H.245 enrutadas a través del gatekeeper.

4.1.3. Información de Depuración

-o -output filename

Escribe anotaciones de trazado hacia un archivo específico.

-t -trace

Ver trazado detallado (verbosity). Cuantas más -t se añadan, más detallada será la salida. Por ejemplo, utilice -ttttt para configurar el nivel de detalle al 5.

4.2. Archivo de Configuración

El archivo de configuración es un archivo estándar de texto. El formato básico es el siguiente:

```
[Section String]
Key Name=Value String
```

Los comentarios están marcados con un signo de numeral(#) o un punto y coma(;) al inicio de cada línea.

El archivo complete.ini contiene todas las secciones disponibles para el GnuGk. En la mayoría de los casos no tiene sentido utilizar todas las secciones al mismo tiempo. El archivo simplemente contiene una colección de ejemplos para muchas configuraciones.

El archivo de configuración puede ser cambiado en tiempo de ejecución. Una vez que haya modificado el archivo de configuración, usted puede emitir el comando reload vía puerto de estado, o enviar una señal HUP al gatekeeper en ambientes UNIX. Por ejemplo,

```
kill -HUP `cat /var/run/gnugk.pid`
```

4.2.1. Sección [Gatekeeper::Main]

- Fortytwo=42
  Default: N/A
  Esta configuración permite probar la existencia del archivo de configuración (gatekeeper.ini o cualquiera que sea su nombre) con el que va a trabajar el gatekeeper. Un mensaje de advertencia se mostrará en caso de que no existiera dicho archivo.

- Name=OpenH323GK
  Default: OpenH323GK
  Identificador del gatekeeper. El gatekeeper responderá solamente a los mensages GRQs con este ID y utilizará éste en los mensajes enviados a sus endpoints.

- Home=192.168.1.1
  Default: 0.0.0.0
  El gatekeeper escuchará por peticiones desde esta dirección IP. Por defecto, el gatekeeper escucha desde todas las interfaces de su equipo. Usted puede omitir esta opción a menos que usted desee que el gatekeeper se enlace solamente hacia una dirección IP específica. Pueden agregarse varias direcciones IP en este campo separadas por un punto y coma (;) o una coma (,).

- NetworkInterfaces=192.168.1.1/24,10.0.0.1/0
  Default: N/A
  Aquí especifique las interfaces de red del gatekeeper. Por defecto el gatekeeper detectará las interfaces de su equipo automáticamente. Existen dos situaciones en las que usted podría usar esta opción. Una es en el caso de que la detección automática fallara. Si usted está utilizando el GnuGk detrás de una NAT box entonces deberá utilizar la coniguración de IP externa la cual configurará automáticamente el GnuGK para que opere como si éste estuviera sobre la NAT box.

- EndpointIDSuffix=_gk1
  Default: _endp
  El gatekeeper asignará un único identificador a cada endpoint registrado. Esta opción puede ser utilizada para especificar un sufijo que será añadido al identificador del endpoint. Esto es muy utilizado solamente cuando se utiliza más de un gatekeeper.

- TimeToLive=300
  Default: -1
  El registro de un endpoint con el gatekeeper puede tener un limitado tiempo de vida. El gatekeeper especifica la duración del registro de un endpoint incluyendo un campo timeToLive en el mensaje RCF. Después del tiempo especificado, el registro expira. Los endpoints enviarán periódicamente un mensaje RRQ con el bit keepAlive establecido antes del tiempo de expiración. Estos mensajes incluyen una mínima cantidad de información como se describe en H.225.0. Éste es conocido como lightweight RRQ.

  Esta opción de configuración especifica el contador time-to-live en segundos hasta que el registro expira. Tenga presente que el endpoint puede requerir un timeToLive muy corto en el mensaje RRQ enviado al gatekeeper. Para evitar una sobrecarga de mensajes RRQ, el gatekeeper ajusta automáticamente este contador a 60 segundos si usted estableció un valor menor.

  Después del tiempo de expiración, el gatekeeper enviará seguidamente dos mensajes IRQ para consultar si el endpoint esta aun ejecutándose. Si el endpoint responde con un mensaje IRR, el registro se extenderá. En otro caso el gatekeeper enviará un mensaje URQ con razón ttlExpired hacia el endpoint. El endpoint debe entonces reintentar el registro con el gatekeeper usando un mensaje RRQ completo.

  Para deshabilitar esta opción, establezca esta opción a -1

- TotalBandwidth=100000
  Default: -1
  Aquí se define el total del ancho de banda que va a ser asignado a los endpoints. Por defecto esta característica está deshabilitada. Sea cuidadoso al usar esta opción, puesto que muchos endpoints tienen defectos de implementación.

- RedirectGK=Endpoints > 100 || Calls > 50
  Default: N/A
  Esta opción le permite redireccionar endpoints hacia gatekeeper alternos cuando el gatekeeper se sobrecarga. Por ejemplo, en la configuración anterior, el gatekeeper rechazará un RRQ si los endpoints registrados sobrepasan los 100 o rechazará un ARQ si las llamadas actuales sobrepasan las 50.

  Además, usted puede redireccionar todos los endpoints de manera explícita configurando esta opción a temporary or permanent. El gatekeeper devolverá

un mensaje de rechazo RAS con una lista de gatekeepers alternos definida en AlternateGks. Tenga presente que un redireccionamiento permanente permanent significa que los endpoints redireccionados no se registrarán nuevamente con este gatekeeper. Además tenga en cuenta que esta función solamente tiene efecto con endpoints que cumplen con el estándar H323 versión 4.

- AlternateGKs=1.2.3.4:1719:false:120:OpenH323GK
  Default: N/A
  Con esta opción nosotros permitimos la existencia de otro gatekeeper para proveer redundancia. Esta opción está implementada de manera activa-activa. Actualmente, usted podría encontrarse en una situación (válida por supuesto) donde algunos endpoints están registrados con el primer gatekeeper y algunos están registrados con el segundo gatekeeper. Usted debe ser capaz de usar los dos gatekeepers en un modo round robin para compartir la carga (load-sharing)(eso no está probado aún). Si continúa leyendo Al hablar de gatekeeper primario ("primary GK") se refiere al gatekeeper que estamos configurando actualmente y gatekeeper alterno (.ªlternate GK") se refiere a otro. El gatekeeper primario incluye un campo en el RFC que le indica al endpoint cual dirección IP e identificación del gatekeeper alterno utilizar. Pero en cambio el gatekeeper alterno necesita conocer todos los registros realizados por el gatekeeper primario o sino este rechazará las llamadas. Por consiguiente nuestro gatekeeper puede reenviar cada RRQ hacia una dirección IP alterna.

  La opción de configuración AlternateGKs hace referencia a los campos contenidos en el mensaje RCF del gatekeeper primario . Los dos primeros campos del string definen hacia donde (ip:port) se va a redireccionar. El tercer campo le indica a los endpoints si ellos necesitan registrarse con el gatekeeper alterno antes de de que tengan lugar las llamadas. Ellos normalmente no lo hacen puesto que nosotros reenviamos sus mensajes RRQ, de esta manera ellos se registran con el gatekeeper alterno también. El cuarto campo especifica la prioridad para este gatekeeper. Si es pequeña, mejor, usualmente el gatekeeper primario es considerado para que tenga prioridad 1. Finalmente el último campo especifica el identificador del gatekeeper alterno.

- SendTo=1.2.3.4:1719
  Default: N/A
  A pesar de que esta información esta incluida en el gatekeeper alterno (AlternateGKs), usted debe especificar hacia que dirección reenviar los mensajes RRQs. Esto podría diferir de las direcciones del gatekeeper alterno (AlternateGKs), de esta manera esta es una opción de configuración aparte (piense en máquinas multihomed).

- SkipForwards=1.2.3.4,5.6.7.8
  Default: N/A
  Para evitar reenvíos circulares, usted no debe reenviar los RRQs que usted obtiene desde otros gatekeepers (Esta declaración es verdadera para ambos el gatekeeper primario y alterno). Dos mecanismos son utilizados para identificar si una petición debería ser reenviada. El primero busca una bandera en los mensajes RRQ. Debido a que pocos endpoints implementan esto, nosotros necesitamos una segunda manera que sea más fiable. Especificar la IP de otro gatekeeper en esta lista.

- StatusPort=7000
  Default: 7000

Este es el puerto de estado para monitorear el gatekeeper. Revisar 13 (esta sección) para más detalle.

- SignalCallId=1
  Default: 0
  IDs de señalización de llamadas ACF/ARJ/DCF/DRJ/RouteRequest en los mensajes del puerto de estado. Revisar 13 (esta sección) para más detalle.

- StatusTraceLevel=2
  Default: 2
  Aquí se especifica el nivel de rastreo de mensajes para nuevas interfaces clientes. Revisar 13 (esta sección) para más detalle.

- TimestampFormat=ISO8601
  Default: Cisco
  Esta opción permite controlar el formato de impresión de las cadenas de fecha y hora generados por el gatekeeper. Esta opción afecta a lon módulos: 9.4 ([SqlAcct]), 9.3 ([RadAcct]), 9.2 ([FileAcct]) y otros módulos, excepto a 12.1 ([CallTable]). Usted puede adaptar de acuerdo a sus necesidades el formato de impresión de los strings configurando manualmente el parámetro TimestampFormat.

  Hay cuatro formatos predefinidos:

  - RFC822 - Formato por defecto utilizado por el gatekeeper (ejemplo: Wed, 10 Nov 2004 16:02:01 +0100)
  - ISO8601 - Formato ISO estándar (ejemplo: 2004-11-10 T 16:02:01 +0100)
  - Cisco - Formato utilizado por equipos Cisco (ejemplo: 16:02:01.534 CET Wed Nov 10 2004)
  - MySQL - Formato simple que puede ser comprendido por MySQL (ejemplo: 2004-11-10 16:02:01)

  Si usted necesita otro formato, puede construir sus propios formatos utilizando reglas de la función conocida de C strftime (Revise man strftime o busque en MSDN por strftime) En general, el formato del string consiste de caracteres regulares y códigos formateados, precedidos por un signo de porcentaje (%). Por ejemplo: "%Y-%m-%d y porcentaje%%"dará como resultado "2004- 11- 10 y porcentaje%". Algunos códigos formateados comúnes son:

  - %a - Nombre abreviado del día de la semana
  - %A - Nombre completo del día de la semana
  - %b - Nombre abreviado del mes
  - %B - Nombre completo del mes
  - %d - Día del mes como un número decimal
  - %H - Hora en el formato 24 horas
  - %I - Hora en el formato 12 horas
  - %m - Mes en número decimal
  - %M - Minuto en número decimal
  - %S - Segundo en número decimal
  - %y - Año sin siglo
  - %Y - Año con siglo

- %u - Microsegundos en número decimal (Ésta es una extensión del gnugk)
- %z - Abreviación de la zona horaria (+0100)
- %Z - Nombre de la zona horaria
- %% - Signo de porcentaje

- EncryptAllPasswords=1
  Default: 0
  Habilita la encriptación de todas las contraseñas en la configuración (SQL passwords, RADIUS passwords, [Password] passwords, [GkStatus::Auth] passwords). Si habilitamos esta opción, todos las contraseñas deben estar encriptados utilizando la utilidad addpasswd. De otra manera sólo las contraseñas de [Password] y [GkStatus::Auth] son encriptadas (antiguo comportamiento).

- KeyFilled=0
  Default: N/A
  Define un byte de relleno global para ser utilizado durante la encriptación/desencriptación de contraseñas. Ésta puede ser sustituída por la configuración KeyFilled establecida dentro de cada sección particular de configuración. Usualmente, usted no necesitará cambiar esta opción.

La mayoría de los usuarios nunca necesitará cambiar cualquiera de los siguientes valores: Estos son utilizados para operaciones de testing o aplicaciones avanzadas.

- UseBroadcastListener=0
  Default: 1
  Define si se escucha o no las peticiones broadcast RAS. Esto requiere enlazarse hacia todas las interfaces en una máquina de tal manera si usted desea ejecutar múltiples instancias de gatekeepers sobre la misma máquina, usted debe deshabilitar esta opción.

- UnicastRasPort=1719
  Default: 1719
  El identificador del canal RAS TSAP para unicast.

- MulticastPort=1718
  Default: 1718
  El identificador del canal RAS TSAP para multicast.

- MulticastGroup=224.0.1.41
  Default: 224.0.1.41
  El grupo multicast para el canal RAS.

- EndpointSignalPort=1720
  Default: 1720
  Puerto por defecto para el canal de señalización de llamadas de endpoints.

- ListenQueueLength=1024
  Default: 1024
  Longitud de la cola para conexiones TCP entrantes.

- SignalReadTimeout=1000
  Default: 1000
  Tiempo en milisegundos para leer la interrupción en canales de señalización de llamadas (Q931).

- StatusReadTimeout=3000
  Default: 3000
  Tiempo en milisegundos para leer la interrupción en canales de estado (status channel).

- StatusWriteTimeout=5000
  Default: 5000
  Tiempo en milisegundos para escribir la interrupción en canales de estado (status channel).

- ExternalIP=myip.no-ip.com
  Default: NA/
  Cuando utilice el GnuGK detrás de una NAT, usted puede establecer la dirección IP externa con la que usted desee enmascarar al GK. Ésto permitirá que EP's externos y otros gatekeepers contacten al GK NATeado. Para que esto funcione, se debe habilitar los puertos requeridos por la IP del GK, o ubicar el GK en la NAT box DMZ.

- ExternalIsDynamic=1
  Default: 0
  Defina si la IP externa es dinámica y donde serán pedidas las consultas para mantener la IP externa actualizada. Para que esto funcione, se debe especificar la IP externa con una dirección DNS mantenida por un servicio DDNS tal como www.dyndns.com o www.no-ip.com.

- DefaultDomain=gnugk.org
  Default: NA/
  Al recibir una petición de una dirección en el formato user@domain.com. Esta opción despojará el dominio de la dirección que corresponda a este valor y procesa la petición como si se tratara solamente de un usuario. Esto es muy cómodo cuando tratamos con llamadas interdominio ubicadas mediante la política "srv routing policy"donde se recibe la URI completa. Esto además puede ser utilizado conjuntamente con la sección [RasSrv::RewriteAlias] para convertir la URI recibida en un número E164 para opciones de enrutado.

4.2.2. Sección [GkStatus::Auth]

En esta sección se definen un número de reglas para determinar quienes están permitidos de conectarse al gatekeeper vía puerto de estado (vía telnet). Quien quiera que tenga acceso al puerto de estado tiene un control completo sobre el gatekeeper. Asegúrese de que ésta sección esté configurada correctamente.

- rule=allow
  Default: forbid
  Posibles valores son:
  - forbid - Niega cualquier conexión.
  - allow - Permite cualquier conexión.
  - explicit - Lee el parámetro ip=value donde ip es la dirección IP del cliente observado, value puede ser 1,0 o allow,forbid o yes,no. Si ip no está en la lista el parámetro default es usado.
  - regex - La IP del cliente se hace corresponder con la expression regular dada.

Ejemplo:

Para permitir clientes desde 195.71.129.0/24 y 195.71.131.0/24:

regex=^195\.71\.(129|131)\.[0-9]+$

- password - El usuario tiene un username y password apropiado para conectarse (login). El formato de username/password es el mismo que el de la sección 8.3 ([SimplePasswordAuth]).

Por otra parte, estas reglas puede ser combinadas por "|.º "&". Por ejemplo,

- rule=explicit | regex
  La IP del cliente debe validarse con las reglas explicit o regex.
- rule=regex & password
  La IP del cliente debe validarse con las reglas regex, y el usuario tiene que conectarse con un username y password.

- default=allow
  Default: forbid
  Esta opción es utilizada solamente cuando rule=explicit.
- Shutdown=forbid
  Default: allow
  Esta opción es utilizada para especificar si se permite o no apagar el gatekeeper vía puerto de estado.
- DelayReject=5
  Default: 0
  Esta opción especifica que tiempo (en segundos) se debe esperar antes de rechazar un username / password inválido para acceder vía puerto de estado.

4.2.3. Sección [GkStatus::Filtering]

Revisar 13.4 (Filtrado del Puerto de Estado) para más detalle.

4.2.4. Sección [LogFile]

En esta sección se define parámetros relacionados con el archivo de log. Actualmente se permite a los usuarios especificar las opciones de rotación.

- Rotate=Hourly | Daily | Weekly | Monthly
  Default: N/A
  Si se utiliza esta opción. El archivo de log rotará basado en esta configuración. La rotación Hourly habilita la rotación una vez por hora, la rotación daily, una vez por día, la Weekly, una vez por semana y la monthly una vez por mes. Una rotación exacta esta determinada por la combinación de las variables RotateDay y RotateTime. Durante la rotación un archivo existente es renombrado con el formato CURRENT_FILENAME.YYYYMMDD-HHMMSS, donde YYYYMMDD-HHMMSS es reemplazada con la fecha actual, se agregan y nuevas líneas a un archivo vacío. Para deshabilitar la rotación, no utilice la opción Rotate o establézcala a 0.

  Ejemplo 1: Rotación cada hora (00:45, 01:45, ..., 23:45):

  [LogFile]
  Rotate=Hourly

```
RotateTime=45
```

Ejemplo 2: Rotación cada día a las 23:00 (11PM):

```
[LogFile]
Rotate=Daily
RotateTime=23:00
```

Ejemplo 3: Rotación cada Domingo a las 00:59:

```
[LogFile]
Rotate=Weekly
RotateDay=Sun
RotateTime=00:59
```

Ejemplo 4: Rotación en el último día de cada mes:

```
[LogFile]
Rotate=Monthly
RotateDay=31
RotateTime=23:00
```

5. Configuración de los Modos Enrutado y Proxy

5.1. Sección [RoutedMode]

Los mensajes de señalización de llamada pueden ser manejados de dos maneras. El primer método es Señalización de llamada en Modo Directo, en el cual los mensajes de señalización de llamada son intercambiados directamente entre los endpoints. El segundo método es Señalización de llamada mediante Gatekeeper. En este método, los mensajes de señalización de llamada son enrutados a través del gatekeeper entre los endpoints. La selección de cual método utilizar es realizada por el gatekeeper.

Cuando se utiliza la señalización de llamadas en Modo Gatekeeper, el gatekeeper puede seleccionar si enrutar o no el canal de control H.245 (H.245 control channel) y los canales lógicos (logical channels).

Caso I.

El gatekeeper no rutea estos canales. El canal de control H.245 (H.245 control channel) y los canales lógicos (logical channels) se establecen directamente entre los endpoints.

Caso II.

El canal de control H.245 (H.245 control channel) se enruta entre los endpoints a través del gatekeeper, mientras que los canales lógicos (logical channels) se establecen directamente entre los endpoints.

Caso III.

El gatekeeper rutea el canal de control H.245 (H.245 control channel), asi como también los canales lógicos (logical channels), incluyendo el RTP/RTCP para audio

y video, y el canal T.120 para datos. En este caso, ningún tipo de tráfico es intercambiado directamente entre los endpoints. Esto es usualmente llamado un Proxy H.323, el mismo que puede ser considerado también como un gateway H.323-H.323.

Esta sección define las opciones de ruteo del modo gatekeeper (Gatekeeper Routed) (Casos I & II). La característica de proxy está definida en la 5.2 (siguiente sección). Todas las configuracones dentro de esta sección son afectadas por el comando reload desde el puerto de estado.

- GKRouted=1
  Default: 0
  Si se habilita o no el ruteo de la señalización en modo gatekeeper (gatekeeper routed signaling mode).
- H245Routed=1
  Default: 0
  Si se enruta también el Canal de Control H.245 a través del gatekeeper. Solamente tendrá efecto si el parámetro GKRouted=1 y el tunneling H.245 está deshabilitado para una llamada. Incluso cuando esta opción está dehabilitada, si Proxy o ProxyForNAT tiene efecto, siempre será enrutado el canal H.245 a través del gatekeeper para las llamadas que están pasando por el proxy.
- CallSignalPort=0
  Default: 1721
  El puerto por donde el gatekeeper realizará la señalización de llamada. El puerto por defecto es 1721. Nosotros no usamos el puerto conocido 1720 para que usted tenga la posibilidad de ejecutar un endpoint H.323 en la misma máquina del gatekeeper. Usted puede establecer este parámetro en 0 para permitirle al gatekeeper seleccionar un puerto arbitrario.
- CallSignalHandlerNumber=2
  Default: 1
  El numero de hilos dedicados al manejo de los canales de señalización/H.245. Usted puede incrementar este número en un gatekeeper fuertemente cargado. Cada hilo puede procesar un mensaje de señalización a la vez, de esta manera incrementar este número incrementará el rendimiento de la llamada. Bajo Windows, existe un límite por defecto de 64 sockets utilizados por un simple hilo de señalización, así cada hilo de señalización está en condiciones de manejar al menos 32 llamadas (con el tunneling H.245 habilitado).
- RtpHandlerNumber=2
  Default: 1
  El número de hilos que manejarán el RTP proxy. Incremente este valor solamente si usted experimenta problemas con el RTP delay o jitter en gatekeeper cargados fuertemente. Debe tener un cuidado especial bajo Windows, los hilos manejadores de RTP están sujetos al mismo límite de 64 sockets como hilos de señalización. Cada hilo RTP está en condiciones de manejar 32 llamadas proxiadas (2 sockets por llamada).
- AcceptNeighborsCalls=1
  Default: 1
  Con esta característica habilitada, el hilo de señalización de llamada aceptará llamadas sin un CallRec preexistente dentro del CallTable, de tal manera que un

endpoint correspondiente a una dirección de destino (destinationAddress) en un mensaje Setup pueda ser encontrado en la tabla de registro (RegistrationTable), y el emisor de la llamada es su vecino o su GK padre. El gatekeeper además utiliza su propia dirección de señalización de llamada (call signalling address) dentro de LCF en respuesta a un LRQ. Eso significa, la señalización de llamada será ruteada hacia el GK2 en llamadas GK-GK. Como resultado, los CDRs en el GK2 pueden mostrar correctamente el tiempo de conexión en lugar de 'unconnected'.

- AcceptUnregisteredCalls=1
  Default: 0
  Con esta característica habilitada, el gatekeeper aceptará llamadas desde cualquier endpoint no registrado. Sin embargo, esto permite riesgos en la seguridad. Tenga cuidado con esto.

- RemoveH245AddressOnTunneling=1
  Default: 0
  Algunos endpoints envían h245Address dentro del UUIE del Q.931 aún cuando h245Tunneling está establecida en TRUE. Esto podría causar problemas de interoperabilidad. Si la opción es TRUE, el gatekeeper removerá las h245Address cuando el indicador h245Tunneling esté en TRUE. Esto forzará a que la parte remota (remote party) permanezca en modo tunnelling.

- RemoveCallOnDRQ=0
  Default: 1
  Con esta opción establecida en off, el gatekeeper no desconectará una llamada si éste recive un DRQ para la llamada. Esto evita que un DRQ alcance un Release Complete. Esto tiene sentido solamente en gatekeeper en Modo Enrutado puesto que en Modo Directo, el mecanismo de señal end-of-call es un DRQ. Cuando utilice la opción call failover, este parámetro debe ser establecido en 0.

- DropCallsByReleaseComplete=1
  Default: 0
  De acuerdo con la recomendación H.323, el gatekeeper podrá colgar una llamada enviando un mensaje RAS DisengageRequest hacia los endpoints. Sin embargo, algunos endpoints mal implementados ignoran este comando. Con esta opción establecida en .ºn", el gatekeeper enviará mensajes Q.931 Release Complete en lugar de mensajes RAS DRQ hacia ambos endpoints para forzar a que ellos terminen la llamada.

- SendReleaseCompleteOnDRQ=1
  Default: 0
  Cuando se cuelga la llamada, el endpoint envía tanto un Release Complete dentro de H.225/Q.931 y un DRQ dentro de los mensajes RAS. Podría ocurrir que el DRQ sea procesado primero, ocasionando que el gatekeeper cierre el canal de señalización de llamadas, previniendo así que el Release Complete sea enviado hacia el otro endpoint. Aunque el gatekeeper cierre el canal TCP hacia el destino, algunos endpoints (por ejemplo Cisco CallManager) no terminan la llamada aún si el canal de señalización de llamada está cerrado. Esto ocasiona que teléfonos se mantengan sonando si el que llama cuelga antes de que el receptor conteste. Establecer este parámetro en 1 produce que el gatekeeper siempre envíe un Release Complete hacia ambos endpoints antes de que cierren la llamada cuando éste recibe un DRQ desde una de la partes.

- SupportNATedEndpoints=1
  Default: 0

Permite a un endpoint detrás de una NAT box registrarse con el gatekeeper. Si ésto se permite, el gatekeeper traducirá las direcciones IP dentro del canal Q.931 y H.245 hacia la IP de la NAT box.

Desde la versión 2.0.2, El GnuGk soporta salida de llamadas NAT (NAT outbound calls) (desde un endpoint detrás de la NAT hacia redes públicas) directamente sin ninguna modificación necesaria de endpoints o de NAT box. Solamente registre el endpoint con el GnuGk y usted puede hacer sus llamadas ahora.

- SupportCallingNATedEndpoints=0
  Default: 1
  Defina si permitir o no a un endpoint detrás de una NAT box que soporta la técnica GnuGK Nat Traversal recibir llamadas. Utilice esta opción para bloquear gateways erróneos que no soporten la técnica "GnuGK Nat Traversal"de manera apropiada ocasionando de alguna manera problemas de audio cuando tratan de llamar hacia el gateway. Las llamadas hacia los gateways devuelven caller inalcanzable (caller unreachable). Para ser efectivo el parámetro "SupportNATedEndpoints"debe estar establecido en 1.

- TreatUnregisteredNAT=1
  Default: 0
  Utilizado conjuntamente con AcceptUnregisteredCalls y SupportNATedEndpoints tratará automáticamente a todas las llamadas no registradas las mismas que no pueden ser determinadas como propias de la NAT sean tratadas como propias de la NAT.

  No todos los endpoints envían "sourceSignalAddress.en el mensaje setup, el cual puede ser utilizado para determinar si un emisor (caller) es NAT. Esto agrega soporte para aquellos que no son.

- ScreenDisplayIE=MyID
  Default: N/A
  Modifica el DisplayIE del Q.931 hacia un valor específico.

- ScreenCallingPartyNumberIE=0965123456
  Default: N/A
  Modifica el CallingPartyNumberIE del Q.931 hacia un valor específico.

- ScreenSourceAddress=MyID
  Default: N/A
  Modifica el campo sourceAddress del elemento UUIE del mensaje Q.931 Setup.

- ForwardOnFacility=1
  Default: 0
  Si se habilita, en recibir Q.931 Facility con razón callForwarded, el gatekeeper enviará el call Setup directamente hacia el endpoint remitido, en lugar de pasar el mensaje de regreso al caller. Si usted tiene endpoints con defectos de implementación que no pueden manejar Q.931 Facility con razón callForwarded, active esta opción. Tenga en cuenta que esta característica no siempre podría trabajar correctamente, puesto que ésta no proporciona ningun recurso de renegociación de capacidades y media channel reopening.

- ShowForwarderNumber=0
  Default: 0
  Defina esta opción si desea reescribir el número de la parte que llama (calling party) al número del que reenvia (forwarder). Esta característica es utilizada para propósitos de billing. Solamente es válida si ForwardOnFacility=1.

- Q931PortRange=20000-20999
  Default: N/A (permitir que el SO asigne los puertos)
  Especifique el rango de números de puerto TCP para canales de señalización Q.931. Tenga presente que el tamaño del rango limita el número de llamadas concurrentes. Asegúrese que este rango sea suficiente para tomar en cuenta el descanso del socket TCP TIME_WAIT antes de que un socket pueda reutilizarse después de que ha sido cerrado. El valor TIME_WAIT puede variar desde 15 segundos hasta pocos minutos, dependiendo del Sistema operativo. Así, si su rango es 2000-2001 y usted hace dos llamadas, las siguientes dos pueden ser hechas después de que transcurra el TIME_WAIT y el sockets pueda ser reutilizado. Lo mismo se aplica a H245PortRange y T120PortRange. Usualmente el TIME_WAIT puede ser deshabilitado en muchos Sistemas Operativos.

- H245PortRange=30000-30999
  Default: N/A (permitir que el SO asigne los puertos)
  Especifique el rango de puertos TCP para los canales de control H.245. Tenga en cuenta que el tamaño del rango podría limitar el número de llamadas concurrentes. Revise los comentarios acerca del descanso de socket TIME_WAIT en la descripción del Q931PortRange.

- SetupTimeout=4000
  Default: 8000
  Tiempo de espera en milisegundos para que un primer mensaje de Setup sea recibido después que el canal de señalización ha sido abierto.

- SignalTimeout=10000
  Default: 30000
  Tiempo de espera en milisegundos para que un canal de señalización sea abierto después de que se ha enviado un mensaje ACF o espera por un mensaje de alerta después de que el canal de señalización ha sido abierto. Esta opción puede ser imaginada como el máximo permitido del valor PDD (Post Dial Delay).

- AlertingTimeout=60000
  Default: 180000
  Tiempo de espera en milisegundos que espera por un mesaje de conexión (Connect) después de que una llamada entró en estado de alerta (Alerting state). Este valor puede ser imaginado como el tiempo máximo del sonido del telefono ringing time".

- TcpKeepAlive=0
  Default: 1 Habilite/Deshabilite la característica keepalive en sockets de señalización (TCP signaling sockets). Esto puede ayudar a detectar canales de señalización inactivos y prevenir llamadas inactivas (dead calls) imposibles de colgar en la tabla de llamadas (call table). Para que esta opción funcione, usted necesita además modificar las configuraciones del sistema para ajustar el "keep alive timeout". Para más detalle vea docs/keepalive.txt.

- TranslateFacility=1
  Default: 0
  Habilite esta opción si usted tiene problemas de interoperabilidad entre endpoints compatibles con H.323v4 y endpoints no compatibles con H.323v4. Esto convierte los mensajes Facility (Facility messages) con razón = transportedInformation en mensajes Facility (Facility messages) con un cuerpo vacío debido a que algunos endpoints no procesan mensajes tunneled H.245 dentro de los mensajes Facility si el cuerpo no

está vacío. Esta conversión solamente se realiza cuando sea necesaria. Si ambos endpoints son v4 o ambos endpoints son previas a v4, no se realiza conversión alguna.

- SocketCleanupTimeout=1000
  Default: 5000
  Define el tiempo que se debe esperar antes de que un socket no utilizado sea cerrado (Si éste aún no ha sido cerrado) y eliminado (su memoria sea desocupada). Si usted utiliza rangos de puerto muy pequeños o pocos puertos (ejemplo: RTPPortRange=2000-2009), usted debe cambiar este valor para conseguir sockets más rápidamente reutilizables.

- ActivateFailover=1
  Default: 0
  Activa el çall failover": Cuando se activa, el GnuGk tratará de buscar otras posibles rutas de destino si la llamada falla en su primera ruta. Actualmente solo pueden ser utilizadas como rutas alternas aquellas rutas disponibles mediante políticas de ruteo de tipo "internal".

  Para contabilizar las llamadas que utilizan failover, Revise el switch SingleFailoverCDR en la sección 12.1 ([CallTable]).

- FailoverCauses=1-15,21-127
  Default: 1-15,21-127
  Define qué código de causa en un ReleaseComplete lanzará el call failover.

- DiableRetryChecks=1
  Default: 0
  Esto desactivará toda comprobación si una llamada fallida ya ha recibido mensajes FastStart o H.245. Advertencia: El uso de este switch permite reintentar más llamadas, pero se corre el riesgo de que algunas de las llamadas reintentadas puedan fallar a causade que el el caller está aun en un estado donde no puede hacer una nueva llamada.

- CalledTypeOfNumber=1
  Default: N/A
  Configura el tipo de números de Called-Party-Number en un valor específico para todas las llamadas (0 - UnknownType, 1 - InternationalType, 2 - NationalType, 3 - NetworkSpecificType, 4 - SubscriberType, 6 - AbbreviatedType, 7 - ReservedType).

- CallingTypeOfNumber=1
  Default: N/A
  Configura el tipo de números de Called-Party-Number en un valor específico para todas las llamadas (0 - UnknownType, 1 - InternationalType, 2 - NationalType, 3 - NetworkSpecificType, 4 - SubscriberType, 6 - AbbreviatedType, 7 - ReservedType).

5.2. Sección [Proxy]

Esta sección define las características del proxy H.323. Esto quiere decir que el gatekeeper ruteará todo el tráfico entre el endpoint emisor y receptor, de esta manera no existe tráfico directamente entre dos endpoints. Por eso ésto es muy utilizado si usted tiene algunos endpoints utilizando IP privadas detrás de una NAT box y algunos endpoints utilizando IP publicas fuera de la NAT box.

El gatekeeper puede hacer proxy para canales lógicos de RTP/RTCP (audio y video) y T.120 (datos). Canales abiertos por procedimientos de conexión rápida (fast-connect) o H.245 tunnelling también son soportados.

Tenga en cuenta que para hacer que el proxy funcione, el gatekeeper debe tener conexión directa hacia ambas redes: la de emisor (caller) y la del receptor (callee).

- Enable=1
  Default: 0
  Define si se habilita o no la función proxy. Usted debe de habilitar el ruteo en modo gatekeeper primero (ver la 5.1 (sección previa)). Usted no tiene que especificar el ruteo H.245. Este será automáticamente utilizado si se requiere.
- InternalNetwork=10.0.1.0/24
  Default: N/A
  Define las redes detrás del proxy. Se permite el uso de múltiples redes internas. El proxy enruta canales solamente de las comunicaciones entre un endpoint en la red interna y una externa. Si usted no especifica esto, todas las llamadas serán llevadas a través del proxy. Si utiliza un GnuGk detrás de una NAT y el parámetro ExternalIP de la sección [Gatekeeper::Main] está configurado, entonces no es obligatorio establecer éste, tal como es auto-detectado al inicio. Utilizatr esta configuración simplemente sustituirá la configuración detectada por defecto.

  Formato:

  InternalNetwork=network address/netmask[,network address/netmask,...]

  La máscara de red (netmask) puede ser expresada en notación decimal punteada o en notación CIDR (prefix length), como se muestra en el ejemplo.

  Ejemplo:

  InternalNetwork=10.0.0.0/255.0.0.0,192.168.0.0/24
- T120PortRange=40000-40999
  Default: N/A (permitir que el SO asigne los puertos)
  Especifica el rango de números de puertos TCP para los canales de datos T.120. Tenga en cuenta que el tamaño del rango podría limitar el número de llamadas concurrentes. Revise los comentarios acerca del tiempo de espera del socket TIME_WAIT en la descripción de Q931PortRange.
- RTPPortRange=50000-59999
  Default: 1024-65535
  Especifica el rango de números de puertos UDP para los canales RTP/RTCP. Tenga en cuenta que el tamaño del rango podría limitar el número de llamadas concurrentes.
- ProxyForNAT=1
  Default: 1
  Si se activa, el gatekeeper pasará por el proxy aquellas llamadas en las cuales uno de los endpoints participantes esta detrás de una NAT box. Esto asegura que el stream RTP/RTCP pueda penetrar dentro de la NAT box sin modificar éste. Sin embargo, el endpoint que se encuentra detrás de la NAT box debe utilizar el mismo puerto para enviar y recibir el stream RTP/RTCP. Si usted trabaja con endpoints con fallas o que no satisfacen la precondición, debe mejor deshabilitar esta característica y permitir que la NAT box remita el stream RTP/RTCP por usted.

- ProxyForSameNAT=0
  Default: 0
  DEDefine si se habilita el proxy para llamadas entre endpoints desde la misma NAT box. Usted no necesita habilitar generalmente esta característica, ya que normalmente los endpoints de la misma NAT box pueden comunicarse entre si.

- EnableRTPMute=1
  Default: 0
  Esta característica permite a la parte que llama (call party) en modo proxy silenciar el audio/video enviando un * como string o como tone.userinput. El envío de * silencia el audio/video y enviar nuevamente un * desactiva el silencio del audio/video. Solamente la parte que silencia la llamada puede desactivar el silecio. Esto esta diseñado como una función "hold"para terminales que no soportan H450.4.

6. Configuración del Enrutado

Las siguientes secciones del Archivo de Configuración son utilizadas para configurar el enrutamiento de las llamadas.

6.1. Sección [RoutingPolicy]

Esta sección explica el funcionamiento de las diferentes políticas de enrutado del gatekeeper.

Las peticiones de llamada entrantes pueden ser encaminadas usando un número de proovedores de ruta:

- explicit
  El destinatario se detalla de manera explícita en la petición de enrutado.

- internal
  La regla clásica; buscar el destinatario en la tabla de registro (RegistrationTable).

- parent
  Enruta la llamada utilizando información obtenida desde el GK padre en respuesta a un mensaje ARQ enviado por el gatekeeper.

- neighbor
  Enruta la llamada usando vecinos a través de mensajes LRQ

- dns
  El destino se obtiene del DNS, dado que es obtenible.

- vqueue
  Usar el mecanismo de cola virtual y generar un evento RouteRequest para permitir manejar el enrutado a una aplicación externa (sólo puede ser usada con OnARQ o OnLRQ)

- numberanalysis
  Provee soporte para el envío de dígito enmascarado (overlapped digit) para mensajes ARQ. Además soporta parcialmente mensajes Setup (no al envio de dígitos superpuestos - solamente validación de longitud de número).

- enum
  ENUM (RFC3761) es un método para utilizar DNS lookup para convertir números reales IDD E164 en información de marcado H323. Los servidores por defecto son e164.voxgratia.net, e164.org y e164.arpa. Para especificar su propio servidor, tiene que detallar la variable de ambiente PWLIB_ENUM_PATH con la dirección de sus servidores ENUM preferidos separados por un punto u coma (;). (Desde la versión 1.8.0 de PWLib soporta la valiabre PWLIB_ENUM_PATH; La versión 1.7.5.2 (Pandora) no soporta ésto.)

  La política .enumremplaza el destino con la información devuelta por el servidor ENUM, así que usted debe tener las políticas de ruteo apropiadas para dirigir finalmente la llamada después de la política .enum". Generalmente usted debe tener la política "dns"después de la política .enum"puesto que la nueva ubicación es devuelta en la forma 'number@gatekeeper' entonces se necesita la política "dns"para resolver ésto.

  Finalmente no olvide que cada chequeo de ruta con la política .enumrequiere un DNS lookup. Para agilizar su ruteo, asegúrese de resolver internamente antes de aplicar la política .enum".

- srv
  DNS SRV o H.323 Anexo Q. permite el enrutado de llamadas utilizando el H.323 URI. La direcciones pueden establecerce como usuario (a) dominio. Las direcciones de señalización H.323 son almacenadas en los registros del dominio DNS. Estas direcciones pueden ser direcciones de señalización (signalling address) o direcciones LRQ (LRQ address).

La siguiente es la configuración por defecto para las políticas de enrutado (routing policies):

```
   [RoutingPolicy]
default=explicit,internal,parent,neighbor
```

Si una política no se cumple, se tratará con la siguiente política.

Estas políticas pueden ser aplicadas a diversos tipos de peticiones de enrutado, y datos de entrada de enrutado. Éstos son los siguientes: ARQ, LRQ, Setup y Facility (con razón callForwarded) Existe también una política general de enrutado, que viene a ser la opción por defecto frente a los otros tipos.

Ejemplo:

```
   [RoutingPolicy]
h323_ID=dns,internal
002=neighbor,internal
Default=internal,neighbor,parent
```

Cuando se recibe uno de los mensajes que requiere una decisión de asignación de ruta todas las llamadas con un alias del tipo h323_ID serán resueltas utilizando DNS. Si el DNS no resuelve el alias, se comprueba con las tablas de registro internas. Si se pide un alias que comience con 002, los vecinos son comprobados antes de las tablas de registro. Si el alias solicitado no es del tipo h323_ID o comienza por 002, se usa la política por defecto consultando las tablas de registro internas, tras ellas, a los vecinos, y, en caso de fallo, al padre.

Para los mensajes ARQ, LRQ, Setup y Facility se deberían usar las secciones [RoutingPolicy::OnARQ],[RoutingPolicy::OnLRQ], [RoutingPolicy::OnSetup] y [RoutingPolicy::OnFacility] utilizando la sintaxis anteriormente explicada.

Ejemplo:

```
    [RoutingPolicy::OnARQ]
default=numberanalysis,internal,neighbor
```

Una configuración típica de enrutado con ENUM podría quedar como lo que sigue:

Ejemplo:

```
    [RoutingPolicy]
default=explicit,internal,enum,dns,internal,parent,neighbor
```

6.2. Sección [RasSrv::RewriteE164]

Esta sección define las reglas de reescritura para números dialedDigits (números E.164)

Formato:

[!]original-prefix=target-prefix

Si el número comienza con original-prefix, se reescribe a target-prefix. Si el comodín '!' precede a original-prefix, se invierte el sentido y el prefijo del objetivo se antepone al número marcado. Se permiten los comodines especiales ('.' y '%').

Ejemplo:

08=18888

Si se marca 08345718, se reescribe a 18888345718.

Ejemplo:

!08=18888

Si se marca 09345718, se reescribe a 1888809345718.

Opción:

- Fastmatch=08
  Default: N/A
  Sólo reescribe dialDigits que comiencen con el prefijo especificado.

6.3. Sección [RasSrv::RewriteAlias]

Esta sección define las reglas de reescritura de alias. Esto puede ser utilizado para mapear los alias asignados por el gatekeeper a los endpoints registrados.

Formato:

[!]original-alias=target-alias

Si el alias es original-alias, éste es reescrito a target-alias.

Ejemplo:

bill=033123456

6.4. Sección [RasSrv::GWRewriteE164]

Esta sección describe la reescritura de números dialedDigits E.164 en función de si llega o se envía la llamada desde el gateway. Esto permite una manipulación más flexible de los dialedDigits para el enrutado. En combinación con el 6.3 (RasSrv::RewriteE164) se puede realizar la reescritura en tres fases:

```
Llamada desde el "gw1", dialedDigits 0867822
                |
                |
                V
Reglas de entrada para "gw1", dialedDigits ahora es 550867822
                |
                |
                V
Reglas globales, dialedDigits ahora es 440867822
                |
                |
                V
Selección de Gateway, dialedDigits ahora es 440867822, Gateway de cara al exterior "gw2"
                |
                |
                V
Reglas de salida para "gw2", dialedDigits ahora es 0867822
                |
                |
                V
Llamada a "gw2", dialedDigits 0867822
```

Formato:

gw-alias=in|out=[!]original-prefix=target-prefix[;in|out...]

Si la llamada re corresponde al gateway, la dirección que comienza con original-prefix se reescribe a target-prefix. Si la bandera '!' precede a original-prefix, el sentido se invierte. Se permiten los caracteres especiales ('.' and '%'). Es preciso separar con ';' las diversas reglas para un mismo gateway.

Ejemplo:

gw1=in=123=321

Si se recibe una llamada desde "gw1.ª 12377897, se reescribe a 32177897 antes de completar cualquier otra acción.

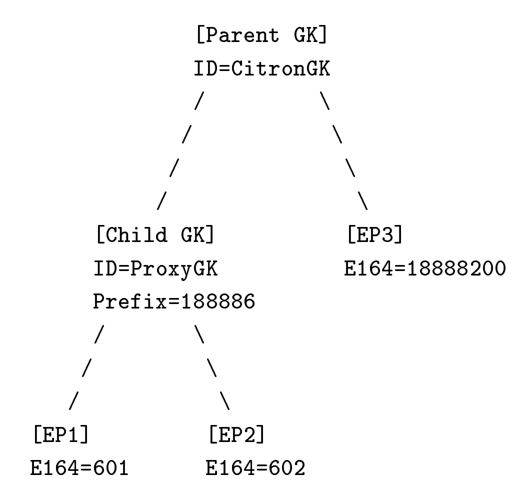
6.5. Sección [Endpoint::RewriteE164]

Una vez especificado el/los prefijos para el gatekeeper que trabaja como endpoint, el gatekeeper padre enrutará las llamadas con dialedDigits que empiezan con esos prefijos. El gatekeeper hijo puede reescribir el destino de acuerdo a las reglas especificadas en esta sección. En contraste, cuando un endpoint interno llama a un endpojnt registrado en el gatekeeper padre, el origen se escribirá a la inversa.

Formato:

external prefix=internal prefix

Por ejemplo, Si se tiene la siguiente configuración:



Con esta regla:

```
188886=6
```

Cuando EP1 llame a EP3 por 18888200, el CallingPartyNumber en el mensaje Setup Q.931 se reescribirá a 18888601. A la inversa, EP3 alcanza EP1 y EP2 llamando a 18888601 y 18888602 respectivamente. Por tanto, un endpoint registrado con el GK hijo con prefijo '6' aparecerá como un endpoint con prefijo '188886', para los endpoints registrados con el gatekeeper padre.

La sección no se relaciona con la sección 6.3 (RasSrv::RewriteE164), aunque la última tendrá efecto antes.

6.6. Sección [Routing::NumberAnalysis]

Esta sección define reglas para la política de enrutado numberanalysis. Esta política revisa el mínimo y/o máximo número de dígitos que tiene un número marcado y envía un mensaje ARJ si es necesario (número de dígitos fuera de rango), permitiendo dar soporte al envío de dígitos superpuestos (overlapped digit). Se da un soporte parcial a los mensajes Setup (no al envio de dígitos superpuestos - solamente a la validación de la longitud del número).

Formato:

prefix=MIN_DIGITS[:MAX_DIGITS]

Si el número empareja con prefix, éste se verifica para ver si consiste de al menos MIN_DIGITS dígitos y (si también se agregó MAX_DIGITS) hasta un máximo de MAX_DIGITS dígitos. Tambien se pueden utilizar caracteres especiales como !, '.' y '%'). Si el número es demasiado corto , se devolverá un mensaje ARJ con el campo rejectReason espablecido a incompleteAddress. Si el número es demasiado largo, se devolverá un mensaje ARJ con el campo rejectReason establecido a undefinedReason. Para buscar una correspondencia se busca en la lista de prefijos desde el prefijo más largo hasta el prefijo mas corto. En el caso de mensajes Setup, se devolverá un Release Complete con "badFormatAddressçuando el número tiene una logitud incorrecta.

Ejemplo:

```
[RoutingPolicy::OnARQ]
default=numberanalysis,internal

[Routing::NumberAnalysis]
0048=12
48=10
.=6:20
```

Las llamadas que van hacia destinatarios que empiezan con 0048 deben tener por lo menos 12 dígitos, los que van hacia destinatarios que empiezan con 48 deben tener 10 dígitos y resto por lo menos 6 y como máximo 20 dígitos.

6.7. Sección [RewriteCLI]

Esta sección contiene un conjunto de reglas de reescritura para números ANI/CLI (caller id). El proceso de reescritura es realizado en dos estados - reescritura entrante (inbound rewrite) y reescritura saliente (outbound rewrite). La reescritura entrante (inbound rewrite) se realiza antes que cualquier otro proceso de mensajes Q.931 (como inbound GWRewrite, autenticación, accounting, ...) y éste tendrá efectos visibles dentro de los módulos auth/acct, puesto que éste afecta al Calling-Station-Id. La reescritura saliente (outbound rewrite) toma lugar justo antes de que el mensaje de Setup sea remitido y su efecto es visible solamente para el endpoint que recibe la llamada (callee).

Una regla de reescritura entrante (inbound rewrite) puede enparejarse con una IP del endpoint emisor (caller's IP) y un número marcado (dialed number) o un CLI/ANI original. Una regla de reescritura saliente (outbound rewrite) puede emparejarse con una IP del endpoint emisor (caller's IP), con la IP del receptor (callee's IP) y con un número marcado (dialed number) o con un número de destino (el número marcado después de la reescritura) o un CLI/ANI (después de la reescritura entrante).

Este módulo además provee la característica CLIR (Calling Line Identification Restriction) que puede ser configurada para cada endpoint (regla).

- ProcessSourceInfo=1
  Default: 1
  Además de reescribir un Calling-Party-Number IE también se podrá reescribir el elemento sourceInfo del mensaje de Setup H.225.0, de esta manera ambos contienen información coherente.

- RemoveH323Id=1
  Default: 1
  Cuando se reescribe el elemento sourceInfo de un mensaje Setup H.225.0, los alias de tipo H323_ID, email_ID y url_ID pueden permanecer intactas si esta opción permanece deshabilitada.

- CLIRPolicy=apply
  Default: N/A
  Aquí puede establecerse un indicador de presentación global que procesa la política. Esta política puede ser aplicada a todas las reglas de reescritura CLI que no anulen a ésta. Las posibles selecciones son forward - solamente remite el PI recibido tal como está, apply - examina el PI recibido y oculta el CLI si éste esta establecido en "presentación restringida" applyforterminals - similar a apply excepto que el

número es removido solamente cuando la llamada es enviada hacia un terminal y no hacia un gateway.

Formato para una regla entrante:

in:CALLER_IP=[pi=[allow|restrict][,forward|apply|applyforterminals]]
[cli:|dno:]number_prefix(=|*=|~=)NEW_CLI[,NEW_CLI]...

El prefijo in: indica que ésta es una regla entrante y el CALLER_IP será utilizado para emparejar la regla (éste puede ser una simple IP o una subred completa).

El parámetro opcional pi= controla las características de CLIR (Calling Line Identification Restriction). Con los valores de allow o restrict establecemos que el indicador de presentación (presentation indicator) sea "presentación permitida.º "presentación restringida". Los indicadores forward, apply y applyforterminals controlan cómo el indicador de presentación recibido (si lo hay) es procesado por el gatekeeper. forward significa reenviar el indicador hacia el callee tal como está, apply significa ocultar el CLI si el PI está establecido en "presentación restringida", applyforterminals es similar a apply, excepto que el CLI se oculta solamente cuando la llamada es enviada hacia un terminal, y no a un gateway.

El prefijo cli: o dno: (el valor predeterminado) selecciona el número que será utilizado para emparejar el number_prefix - un id de caller (CLI/ANI) o un número marcado (dialed number). El emparejado/reescritura de números puede realizarse de tres maneras:

- = - Un número cli o dno se emparejará contra el number_prefix, utilizando un prefijo (prefix match), si el prefijo es encontrado, el CLI será reemplazado con el NEW_CLI,
- ~= - Un número cli o dno se emparejará contra el number_prefix, utilizando una indetidad (identity match), si ambos números son los mismos, el CLI será reemplazado con el NEW_CLI,
- *= - (VALIDO SOLAMENTE PARA cli) Un número cli se emparejará contra el number_prefix, utilizando un prefijo (prefix match), si el prefijo es encontrado, el prefijo emparejado CLI (number_prefix) será reemplazado con un prefijo NEW_CLI.

Después del signo de igualdad (=/ =/*=), sigue una lista de valores nuevos de CLI a ser usados. Si se especifica mas de un valor, un único valor se escogerá de manera aleatoria. Está permitido especificar grandes rangos de números como 49173600000-49173699999 (para rangos de números CLIs debe fijarse una longitud fija). Hay una cadena especial constante .ªny", que puede ser utilizada en lugar del CALLER_IP o el number_prefix. Para habilitar la característica CLIR para esta regla, utilice la cadena especial constante "hide"

en lugar de la lista de nuevos valores de CLI. Tenga presente que CLIR es más utilizado para reglas salientes (outbound rules).

Ejemplo 1:

```
[RewriteCLI]
in:192.168.1.1=dno:5551=3003
in:192.168.1.1=cli:1001=2222
in:192.168.1.1=any=1111
```

Estas reglas indican que para aquellas llamadas que vienen desde la IP 192.168.1.1: (1) si el usuario marca un número que empieza con 5551, el CLI se establece en 3003, (2) si la llamada es de un usuario que empieza con CLI en 1001, el CLI se establece en 2222, (3) para el resto de llamadas que vienen de dicha IP, el CLI se establece en 1111.

Ejemplo 2:

```
[RewriteCLI]
in:192.168.1.0/24=any=18001111
in:192.168.2.0/24=any=18002222
in:any=any=0
```

Estas reglas indican que: (1) para las llamadas que vienen desde la red 192.168.1.0/24, el CLI se establece en 18001111, (2) para las llamadas que vienen desde la red 192.168.2.0/24, el CLI se establece en 18002222, (3) para el resto de llamadas, el CLI se establece en 0.

Ejemplo 3:

```
[RewriteCLI]
%r1% in:192.168.1.0/24=0048*=48
%r2% in:192.168.1.0/24=0*=48
in:any=100.~=48900900900
```

Estas reglas indican que: (1) para las llamadas que vienen desde la red 192.168.1.0/24, reescriba los números que empiezan con 0048 a 48 (ejemplo - 0048900900900 => 48900900900), (2) para las llamadas que vienen desde la red 192.168.1.0/24, reescriba los números que empiezan con 0 a 48 (ejemplo - 0900900900 => 48900900900), (3) para el resto de llamadas, si CLI es de 4 dígitos y empieza con 100, establezca éste en 48900900900.

Ejemplo 4 (CLIR):

```
[RewriteCLI]
in:192.168.1.0/24=any=hide
```

Este ejemplo provoca que el número del emisor (caller) sea removido de los mensajes de Setup originados en la red 192.168.1.0/24. Esto además provoca que sean establecidos de manera apropiada los indicadores de presentación y screening en los mensajes Setup.

Formato para la regla saliente (outbound rule):

out:CALLER_IP=CALLEE_IP [pi=[allow|restrict][,forward|apply|applyforterminals]] [cli:|dno:|cno:]number_prefix(=|~=|*=)NEW_CLI[,NEW_CLI]...

El prefijo out: indica que ésta es una regla saliente (outbound rule), el CALLER_IP y el CALLEE_IP serán utilizados para emparejar la regla y puede ser una dirección IP simple o una dirección completa de red.

El parámetro opcional pi= controla las características de CLIR (Calling Line Identification Restriction). Con los valores de allow o restrict establecemos que el indicador de presentación (presentation indicator) sea "presentación permitida.º "presentación restringida". Los indicadores forward, apply y applyforterminals

controlan cómo el indicador de presentación recibido (si lo hay) es procesado por el gatekeeper. forward significa reenviar el indicador hacia el callee tal como está, apply significa ocultar el CLI si el PI está establecido en "presentación restringida", applyforterminals es similar a apply, excepto que el CLI se oculta solamente cuando la llamada es enviada hacia un terminal, y no a un gateway.

El prefijo cli:, dno: (el valor predeterminado) o cno: selecciona el número que será utilizado para emparejar el number_prefix - un id de caller (CLI/ANI), un número marcado (dialed number) o un número destino/called (el número marcado después de la reescritura). El emparejado/reescritura de números puede realizarse de tres maneras:

- = - Un número cli o dno se emparejará contra el number_prefix, utilizando un prefijo (prefix match), si el prefijo es encontrado, el CLI será reemplazado con el NEW_CLI,
- ˜= - Un número cli o dno se emparejará contra el number_prefix, utilizando una indetidad (identity match), si ambos números son los mismos, el CLI será reemplazado con el NEW_CLI,
- *= - (VALIDO SOLAMENTE PARA cli) Un número cli se emparejará contra el number_prefix, utilizando un prefijo (prefix match), si el prefijo es encontrado, el prefijo emparejado CLI (number_prefix) será reemplazado con un prefijo NEW_CLI.

Después del signo de igualdad (=/ =/*=), sigue una lista de valores nuevos de CLI a ser usados. Si se especifica mas de un valor, un único valor se escogerá de manera aleatoria. Está permitido especificar grandes rangos de números como 49173600000-49173699999. Hay una cadena especial constante .ªny", que puede ser utilizada en lugar del CALLER_IP, el CALLEE_IP o el number_prefix. Para habilitar la característica CLIR para esta regla, utilice la cadena especial constante "hide"

o "hidefromterminals"

en lugar de la lista de nuevos valores de CLI.

Ejemplo 1:

```
[RewriteCLI]
out:any=192.168.1.1 any=1001
out:any=192.168.1.2 any=1002
```

Estas reglas establecen un ANI/CLI fijo para cada IP final: (1) presénteme con un ANI 1001, al enviar las llamadas a la IP 192.168.1.1, (2) presénteme con un ANI 1002, al enviar las llamadas a la IP 192.168.1.2.

Ejemplo 2:

```
[RewriteCLI]
out:any=192.168.1.1 any=1001-1999,3001-3999
```

Esta regla selecciona de manera aleatoria un ANI/CLI de un rango de 1001-1999, 3001-3999 para llamadas enviadas a 192.168.1.1.

Ejemplo 3 (CLIR):

```
[RewriteCLI]
out:any=any any=hidefromterminals
out:192.168.1.1=any any=hide
```

En este ejemplo cada suscriptor ha habilitado la característica CLIR. De esta manera todas las llamadas hacia los terminales tendrán un número de emisor (caller) removido y establecidos los indicadores de presentación/screening. Las llamadas que van hacia los gateways tendrán solamente un indicador de presentación establecido en "presentación restringidaz no será removido el número de emisor (caller) para permitir un ruteo apropiado y removimiento de número al equipo destino.
Una excepción a estas reglas es que las llamadas que vienen desde 192.168.1.1 tendrán siempre removido el número de emisor (caller), sin importar si está llamando a un terminal o a un gateway.

Ejemplo 4 (CLIP):

```
[RewriteCLI]
out:any=192.168.1.1 any=hide
```

En este ejemplo la característica CLIP (Calling Line Identification Presentation) está deshabilitada para el usuario 192.168.1.1.

Ejemplo 5 (CLIR):

```
[RewriteCLI]
out:192.168.1.1=any pi=restrict,apply cli:.*=.
out:any=any pi=allow cli:.*=.
```

Estas reglas no cambian el CLI (.*=.) y: (1) habilitan el CLIR para el endpoint 192.168.1.1. apply le indica al gatekeeper que no solamente establezca el PI, sino que además oculte el número actual, (2) Forza la presentación CLI para el resto endpoints.

El emparejamiento de reglas tiene un orden estrictamente definido:

1. Se determina la pareja de la IP del emisor más cercano - closets caller's IP (closest significa con la máscara de red más larga - IPs simples tienen la prioridad más alta, .ªny"tiene la prioridad más baja),
2. (reglas salientes) Se determina la pareja de la IP del receptor más cercano - closest callee's IP,
3. Se busca el prefijo/numero más largo que empareje con el par IP/IP dado, en el siguiente órden:
   a) dno: Se buscan reglas de tipo (número marcado - dialed number),
   b) cno: Se buscan reglas de tipo (número destino/llamado - destination/called number),
   c) cli: Se buscan reglas de tipo (caller id).

Después que ha sido encontrada una pareja para la IP del caller/callee, no se revisa ninguna otra regla, aun cuando ningún prefijo/número se empareja dentro del conjunto de reglas para éstas IPs.

Sobre plataforma Windows, hay un problema con config keys duplicadas, de esta manera hay un workaround para esta restricción. Este ejemplo no podrá trabajar debido a la misma clave (key) (in:192.168.1.1):

```
[RewriteCLI]
in:192.168.1.1=1001=2001
in:192.168.1.1=any=2000
```

Como workaround, puede utilizar una cadena con un signo de porcentaje (%) al inicio y al final antes de la clave (key). Este prefijo se separará automáticamente del nombre de la clave (key name) antes de la carga de las reglas:

```
[RewriteCLI]
%r1% in:192.168.1.1=1001=2001
%r2% in:192.168.1.1=any=2000
```

7. Configuración RAS

7.1. Sección [RasSrv::GWPrefixes]

Esta sección lista qué números E.164 son ruteados hacia un gateway específico.

Formato:

gw-alias=prefix[,prefix,...]

Tenga en cuenta que usted tiene que especificar el alias del gateway. Si un gateway está registrado con un alias, todos los números que empiezan con los prefijos dados son ruteados hacia este gateway. Los caracteres especiales . y ! pueden ser utilizados aqui para hacer corresponder cualquier dígito y deshabilitar el prefijo.

Ejemplo:

test-gw=02,03

7.2. Sección [RasSrv::PermanentEndpoints]

En esta sección usted puede poner endpoints que no tienen soporte RAS o que usted desea que el registro de éstos no expire. Los registros siempre se mantendrán en la tabla de registro del gatekeeper. Sin embargo, usted puede tambien des-registrar estos endpoints desde el puerto de estado. Los caracteres especiales . y ! pueden ser utilizados con prefijos aqui para hacer corresponder cualquier dígito y deshabilitar el prefijo.

Formato:

IP[:port]=alias[,alias,...;prefix,prefix,...]

Ejemplo:

Para gateways,

```
10.0.1.5=Citron;009,008
```

Para terminales,

```
10.0.1.10:1720=700
```

7.3. Sección [RasSrv::RRQFeatures]

- AcceptEndpointIdentifier=1
  Default: 1
  Aqui se define si aceptar o no el endpointIdentifier especificado en un RRQ completo.

- AcceptGatewayPrefixes=1
  Default: 1
  Un gateway puede registrar sus prefijos con el gatekeeper incluyendo supportedPrefixes dentro del campo terminalType del RRQ. Esta opción define si se acepta los prefijos específicos de un gateway.

- OverwriteEPOnSameAddress=1
  Default: 0
  En algunas redes una dirección ip del endpoint puede cambiar inexplicablemente. Esto puede pasar cuando un endpoint está usando conexión PPP (ejemplo: modem o ADSL). Esta opción define cómo manejar una petición de registro (RRQ) desde una dirección IP, la misma que no se corresponde con la que nosotros tenemos almacenada. La acción por default es rechazar la petición. Con esta opción habilitada la petición en conflicto provocaría una petición de des-registro (URQ) para que la dirección IP existente y la entrada sean removidas permitiendo que el endpoint se registre con la nueva dirección.

- IRQPollCount=0
  Default: 1
  Cuando el gatekeeper no recibe un keep-alive RRQ desde el endpoint dentro del período de tiempo TimeToLive, el gatekeeper envía un mensaje IRQ para registrar el estado ("poll") del endpoint y revisa si éste está activo. Después de que sean enviados los mensajes IRQPollCount y no se obtenga una respuesta, el endpoint es des-registrado. Para deshabilitar esta característica (y des-registrar los endpoints inmediatamente después de que el contador TimeToLive termine), establezca esta variable en 0. El intervalo IRQ poll es 60 segundos.

- SupportDynamicIP=1
  Default: 0
  Cuando la dirección IP de un endpoint cambie, el gatekeeper puede mantener el registro del mismo. Esto forzará al EndPoint a registrarse nuevamente de manera completa si la dirección IP cambia.

7.4. Sección [RasSrv::ARQFeatures]

- ArjReasonRouteCallToSCN=0
  Default: 1
  Si es ¿es"(si), el gatekeeper rechaza una llamada que viene desde un gateway hacia si mismo con razón routeCallToSCN.

- ArjReasonRouteCallToGatekeeper=1
  Default: 1
  Si es ¿es", el gatekeeper rechaza un ARQ respondido sin que haya un pre-existente CallRec encontrado en la CallTable con razón routeCallToGatekeeper en modo ruteo. El endpoint liberará la llamada imediátamente y reenviará un call Setup hacia el gatekeeper.

- CallUnregisteredEndpoints=0
  Default: 1
  Con esta opción habilitada, el gatekeeper aceptará un ARQ de un endpoint registrado con destCallSignalAddress, sin importar si la dirección le corresponde a un endpoint registrado o no. Eso significa que usted puede explícitamente especificar la IP del endpoint (registrado o no) que usted desee llamar.
- RemoveTrailingChar=#
  Default: N/A
  Especifique el caracter de rastreo a ser removido en destinationInfo. Por ejemplo, si su endpoint contiene incorrectamente el caracter de terminación como '#' dentro destinationInfo, usted puede remover éste mediante esta opción.
- RoundRobinGateways=0
  Default: 1
  Habilite/Deshabilite la selección de un gateway mediante round-robin, si más de un gateway se emparejan con un número marcado. Si se deshabilita, el primer gateway disponible será seleccionado. De otra manera, las llamadas subsiguientes serán enviadas hacia los gateways utilizando el modo round-robin.

7.5. Sección [NATedEndpoints]

El gatekeeper puede automáticamente detectar si un endpoint está detrás de una NAT. Sin embargo, si la detección falla, usted puede especificar éste manualmente en esta sección.

Formato:

alias=true,yes,1,...

Ejemplo:

Especifique que un endpoint con alias 601 está detrás de la NAT.

```
601=true
```

8. Configuración de la Autenticación

Las siguientes secciones en el Archivo de Configuración pueden ser utilizadas para configurar la Autentificación.

8.1. Sección [Gatekeeper::Auth]

Esta sección define el mecanismo de autenticación para el gatekeeper.

Sintaxis:

```
authrule=acciones

 <authrule> := SimplePasswordAuth | AliasAuth | FileIPAuth | PrefixAuth | RadAuth | RadAliasAuth | SQ
 <acciones>  := <control>[;<ras>|<q931>,<ras>|<q931>,...]
 <control>  := optional | required | sufficient
 <ras>      := GRQ | RRQ | URQ | ARQ | BRQ | DRQ | LRQ | IRQ
 <q931>     := Setup | SetupUnreg
```

Una regla puede resultar en uno de los siguientes tres códigos: ok, fail, pass.

- ok - La petición es autenticada por este módulo.
- fail - La autenticación falla y debe ser rechazada.
- next - La regla no puede determinar la petición.

También hay tres maneras de controlar una regla:

- optional - Si la regla no puede determinar la petición, ésta es enviada a la siguiente regla.
- required - La petición debe ser autenticada por este módulo, de lo contrario sería rechazada. La petición autenticada entonces sería enviada hacia la siguiente regla.
- sufficient - Si la petición es autenticada, ésta es aceptada, de lo contrario sería rechazada. Es decir, la regla determina la suerte de la petición. Ninguna regla debe ser ubicada después de una regla sufficient, puesto que no tendrá ningún efecto.

Módulos soportados actualmente:

- SimplePasswordAuth/SQLPasswordAuth Estos módulos revisan los campos tokens o cryptoTokens de un mensaje RAS. Los tokens deben contener al menos un generalID y un password. Para aquellos tokens cryptoTokens, cryptoEPPwdHash separados (hashed) por MD5 simple y aquellos tokens nestedcryptoToken separados (hashed) por HMAC-SHA1-96 (libssl debe ser instalada) hay soporte actualmente. Para tokens separados (hashed) por CAT (Cisco Access Token) y un username/password en texto claro son soportados actualmente. El ID y password son leídos desde la sección 8.3 ([SimplePasswordAuth]), y desde una base de datos SQL para los módulos SimplePasswordAuth y SQLPasswordAuth. El módulo MySQLPasswordAuth también es soportado para compatibilidades con versiones anteriores.
- AliasAuth/SQLAliasAuth Este módulo puede ser utilizado solamente para auntenticar RegistrationRequest (RRQ). La dirección IP de un endpoint con un alias dado deben corresponderse con un patrón específico. Para AliasAuth el patrón está definido en la sección 8.5 ([RasSrv::RRQAuth]). Para SQLAliasAuth, el patrón es recuperado desde una base de datos SQL, que está definida en la sección 8.6 ([SQLAliasAuth]).
- FileIPAuth Este módulo provee una manera simple de restringir el acceso al gatekeeper basándose en la dirección IP o red del emisor (caller).
- PrefixAuth La dirección IP o alias de una petición con un prefijo dado deben corresponderse o emparejarse con un patrón específico. Revisar la sección 8.8 ([PrefixAuth]) para más detalle. Actualmente el módulo puede autorizar solamente AdmissionRequest (ARQ) y LocationRequest (LRQ).
- RadAuth Provee autenticación basada en el esquema de seguridad H.235 username/password. Autentica RRQ, ARQ y Q.931 Setup mediante servidores RADIUS remotos. Envía al Servidor RADIUS los usernames y passwords extraídos desde los tokens CAT (Cisco Access Tokens) que viajan dentro de los paquetes RRQ, ARQ o Setup. Por consiguiente si sus endpoints no soportan CATs o usted no necesita el esquema de autenticación basado en usernames/password asignados individualmente - este módulo no le servirá (pero puede chequear además el módulo RadAliasAuth). Revisar la sección 8.9 ([RadAuth]) para más detalle.

- RadAliasAuth Provee autenticación basada en alias de endpoints y/o señalización de llamada IP dirigida con servidores remotos RADIUS. Esto no necesita ningún tokens H.235 dentro de los mensajes RAS, de esta manera esto puede ser utilizado sobre una amplia variedad de sistemas de igual manera como RadAuth. Los mensajes RRQ, ARQ y Q.931 Setup pueden ser autentificados utilizando este módulo. Revisar la sección 8.10 ([RadAliasAuth]) para más detalle.
- SQLAuth Un potente módulo para autenticar y autorizar mensajes RRQ, ARQ, LRQ y Setup Con este módulo se podrá realizar verificación basada en varios parámetros, como número de emisor (caller's number), número destino (destination number), username y más. Además tiene soporte para forzar el límite de duración de llamada, reescritura de números, ruteo de llamada, verificación y asignación de alias. Revisar la sección 8.7 ([SQLAuth]) para más detalle.
- CapacityControl Un módulo flexible para controlar el volúmen de llamadas entrantes con la capacidad de configurar varias condiciones. IMPORTANTE: Este módulo tiene que ser utilizado conjuntamente con el módulo de accounting CapacityControl. Revisar la sección 8.11 ([CapacityControl]) para más detalle.

Usted puede además configurar una regla para chequear solamente algunos mensajes RAS particulares. El siguiente ejemplo configura SimplePasswordAuth como regla opcional para chequear mensajes RRQ y ARQ. Si un mensaje RRQ no es chequeado (no contiene campos tokens o cryptoTokens), éste es chequeado por AliasAuth. La regla por defecto es aceptar todas las peticiones.

Ejemplo 1:

```
SimplePasswordAuth=optional;RRQ,ARQ
AliasAuth=sufficient;RRQ
```

El ejemplo mostrado más abajo autentica todas las llamadas, chequeando los mensajes de señalización Setup de una manera detallada, utilizando el módulo RadAliasAuth.

Ejemplo 2:

```
RadAliasAuth=required;Setup
default=allow
```

Este ejemplo chequea los mensajes de registro de endpoints (RRQ) y mensajes de admission de llamadas (ARQ) por cualquiera de estos dos medios: por medio de un username/password (RadAuth) o mediante alias/IP (RadAliasAuth). Adicionalmente, si la llamada es desde un endpoint no registrado (y por consiguiente ninguna de las autenticaciones RRQ o ARQ ha sido realizada), La autenticación con mensajes Setup usando RadAliasAuth toma lugar (SetupUnreg).

Ejemplo 3:

```
RadAuth=optional;RRQ,ARQ
RadAliasAuth=required;RRQ,ARQ,SetupUnreg
default=allow
```

8.2. Sección [FileIPAuth]

Esta sección define una lista de direcciones IP o redes que están permitidas acceder a los recursos del gatekeeper. Puede especificarse una lista de prefijos permitidos junto con una dirección IP. Los eventos que soportan Gatekeeper::Auth son: GRQ, RRQ, LRQ, Setup and SetupUnreg. El formato de una entrada simple es:

IP=[allow | reject][;prefix[,prefix...]]

Donde IP es una dirección IP simple, una dirección de red (en formato A.B.C.D/M.M.M.M o A.B.C.D/LONGITUD) o una cadena 'any' o '*' para emparejar cualquier dirección. La lista de acceso puede además ser cargada desde un archivo externo empleando la directiva include. Durante la autenticación, la longitud de la máscara de red define una prioridad para cada entrada, de esta manera la regla 192.168.1.1=allow se antepone sobre la regla 192.168.1.0/24=reject.

Ejemplo #1:

```
[Gatekeeper::Auth]
FileIPAuth=required;RRQ,LRQ,Setup

[FileIPAuth]
192.168.1.240=reject
192.168.1.0/24=allow
192.168.2.0/255.255.255.0=allow;48,49,44
any=reject
```

Ejemplo #2:

```
[Gatekeeper::Auth]
FileIPAuth=required;Setup

[FileIPAuth]
include=/etc/gnugk/accesslist.ini

(EOF)

Contenido del archivo /etc/gnugk/accesslist.ini:

[FileIPAuth]
192.168.1.1=allow
192.168.1.100=allow
any=reject
```

8.3. Sección [SimplePasswordAuth]

Esta sección define el par de üseridz "passwordütilizados por el módulo SimplePasswordAuth. Todos los passwords están encriptados utilizando la herramienta addpasswd.

Modo de uso:

```
addpasswd config section userid password
```

Ejemplo:

```
addpasswd config.ini SimplePasswordAuth frank secret
```

Opciones:

- KeyFilled=123
  Default: 0
  Valor por defecto para utilizar un byte de relleno durante la encriptación/desencriptación de passwords.
- CheckID=1
  Default: 0
  Verifica si el alias se corresponde con el ID en los tokens.
- PasswordTimeout=120
  Default: -1
  El módulo SimplePasswordAuth y todos sus descendientes mantendrán en caché los passwords autenticados. Este campo define el valor del tiempo de caché en segundos 0 significa que no se utilizará caché para passwords, mientras que un valor negativo significa que la caché nunca expira.

8.4. Sección [SQLPasswordAuth]

Autentica endpoints habilitados con H.235 utilizando passwords almacenadas en una base de datos SQL. Esta sección define el driver SQL a utilizar, los parámetros de connexion a la base de datos y las consultas a utilizar para recuperar los passwords.

- Driver=MySQL | PostgreSQL| Firebird
  Default: N/A
  El driver de la base de datos SQL a utilizar. Actualmente, drivers para tt/MySQL/, PostgreSQL y Firebird están implementados.
- Host=DNS[:PORT] | IP[:PORT]
  Default: localhost
  Dirección del host servidor SQL. Puede estar en la forma de DNS[:PORT] o IP[:PORT]. Como sql.mycompany.com o sql.mycompany.com:3306 o 192.168.3.100.
- Database=billing
  Default: billing
  El nombre de la base de datos a la cual se realiza la conexión.
- Username=gnugk
  El nombre de usuario utilizado para conectarse a la base de datos.
- Password=secret
  El password utilizado para conectarse a la base de datos. Si el password no está especificado, se realiza un intento de conexión a la base de datos sin un password. Si EncryptAllPasswords está habilitada, o la variable KeyFilled esta definida en esta sección, el password esta en forma encriptada y puede ser creado utilizando la herramienta addpasswd.
- CacheTimeout=120
  Default: 0
  Este campo define el tiempo en que el par (alias;password) recuperados desde la base

de datos estarán cacheados en la memoria local. Este valor es expresado en segundos. 0 significa no cachear las passwords, mientras que un valor negativo significa que la caché nunca expira (solamente el comando reload refrescará la caché).

- MinPoolSize=5
  Default: 1
  Define el número de conexiones activas SQL. Esto permite un mejor rendimiento, puesto que más de una consulta puede ser ejecutada al mismo tiempo. MinPoolSize=1 simula un viejo comportamiento, cuando el acceso a la base de datos es serializada (una consulta a la vez).

- Query=SELECT ...
  Default: N/A
  Define la consulta SQL utilizada para recuperar passwords H.235 desde la base de datos. La consulta es parametrizada, esto significa que el reemplazo de parámetros es realizado antes de que cada consulta es ejecutada. Los parámetros están denotados por cadenas %1, %2, ... Especifique %% para insertar un caracter de porcentaje antes de un dígito dentro de un string (así %%1), especifique %{1} para permitir expansión dentro de expresiones complejas como %{1}123. Para SQLPasswordAuth se han definido dos parámetros:

  - %1 - El alias actual para el cual se consultará el password.
  - %2 - El identificador del gatekeeper.

  Aquí algunas cadenas de ejemplo:

  ```
  SELECT h235password FROM users WHERE alias = '%1' AND active
  SELECT h235password FROM users WHERE alias = '%1' AND gk = '%2'
  ```

8.5. Sección [RasSrv::RRQAuth]

Especifica la acción realizada sobre los mensajes de recepción RRQ (confirm or deny) para el módulo AliasAuth. El primer alias (éste será principalmente un H323ID) del endpoint a ser registrado es buscado en esta sección. Si un parámetro es encontrado, el valor sera aplicado por regla general. Una regla consiste de condiciones separadas por "&". Un registro es aceptado cuando todas las condiciones coinciden.

Sintaxis:

```
<authrules> :=  empty  |  <authrule> "&" <authrules>

  <authrule>  := <authtype> ":" <authparams>
  <authtype>  := "sigaddr" | "sigip"
  <autparams> := [!&]*
```

La notación y significado de <authparams> depende de <authtype>:

- sigaddr - Expresión regular extendida que tiene que corresponderse contra la representación "PrintOn(ostream)" de la petición. Ejemplo

  ```
  sigaddr:.*ipAddress .* ip = .* c0 a8 e2 a5 .*port = 1720.*
  ```

- sigip - Forma especializada de 'sigaddr'. Escribe la dirección ip señalizada utilizando la notación decimal (comunmente utilizada): "byteA.byteB.byteC.byteD:port". Ejemplo:

sigip:192.168.242.165:1720

- allow - Siempre aceptar el alias.
- deny - Siempre rechazar el alias.

8.6. Sección [SQLAliasAuth]

Autentica endpoints utilizando reglas almacenadas en una base de datos (Las reglas tienen el mismo formato definido en la sección 8.5 ([RasSrv::RRQAuth])). Esta sección define el driver SQL a ser utilizado, los parámetros de conexión a la base de datos y las consultas a utilizar para recuperar los patroneS.

- Driver=MySQL | PostgreSQL | Firebird
  Default: N/A
  El driver SQL a utilizar. Actualmente, drivers MySQL, PostgreSQL y Firebird están implementados.
- Host=DNS[:PORT] | IP[:PORT]
  Default: localhost
  Dirección del host servidor SQL. Puede estar en la forma de DNS[:PORT] o IP[:PORT]. Como sql.mycompany.com o sql.mycompany.com:3306 o 192.168.3.100.
- Database=billing
  Default: billing
  El nombre de la base de datos a la cual se realiza la conexión.
- Username=gnugk
  El nombre de usuario utilizado para conectarse a la base de datos.
- Password=secret
  El password utilizado para conectarse a la base de datos. Si el password no está especificado, se realiza un intento de conexión a la base de datos sin un password. Si EncryptAllPasswords está habilitada, o una variable KeyFilled esta definida en esta sección, el password está en forma encriptada y puede ser creado utilizando la herramienta addpasswd.
- CacheTimeout=120
  Default: 0
  Este campo define el tiempo en que el par (alias;authrule) recuperados desde la base de datos estarán cacheados en la memoria local. Este valor es expresado en segundos. 0 significa no cachear las passwords, mientras que un valor negativo significa que la caché nunca expira (solamente el comando reload refrescará la caché).
- MinPoolSize=5
  Default: 1
  Define el número de conexiones activas SQL. Esto permite un mejor rendimiento, puesto que más de una consulta puede ser ejecutada al mismo tiempo. MinPoolSize=1 simula un viejo comportamiento, cuando el acceso a la base de datos es serializada (una consulta a la vez).
- Query=SELECT ...
  Default: N/A
  Define la consulta SQL utilizada para recuperar alias desde la base de datos.

La consulta es parametrizada, esto significa que el reemplazo de parámetros es realizado antes de que cada consulta es ejecutada. Los parámetros están denotados por cadenas %1, %2, ... Especifique %% para insertar un caracter de porcentaje antes de un dígito dentro de un string (así %%1), especifique %{1} para permitir expansión dentro de expresiones complejas como %{1}123. Para SQLAliasAuth se han definido dos parámetros:

- %1 - El alias actual para el cual se consultará el password.
- %2 - El identificador del gatekeeper.

Aquí algunas consultas de ejemplo:

```
SELECT authrule FROM users WHERE alias = '%1' AND active
SELECT 'sigip:' || host(ip) || port FROM users WHERE alias = '%1'
```

8.7. Sección [SQLAuth]

Autentica y autoriza endpoints o llamadas utilizando una base de datos SQL. Permite soporte para eventos RRQ, ARQ, LRQ y Setup.

- Driver=MySQL | PostgreSQL | Firebird
  Default: N/A
  El driver de base de datos SQL a utilizar. Actualmente, drivers MySQL, PostgreSQL y Firebird están implementados.

- Host=DNS[:PORT] | IP[:PORT]
  Default: localhost
  Dirección del host servidor SQL. Puede estar en la forma de DNS[:PORT] o IP[:PORT]. Como sql.mycompany.com o sql.mycompany.com:3306 o 192.168.3.100.

- Database=billing
  Default: billing
  El nombre de la base de datos a la cual se realiza la conexión.

- Username=gnugk
  El nombre de usuario utilizado para conectarse a la base de datos.

- Password=secret
  El password utilizado para conectarse a la base de datos. Si el password no está especificado, se realiza un intento de conexión a la base de datos sin un password. Si EncryptAllPasswords está habilitada, o una variable KeyFilled esta definida en esta sección, el password está en forma encriptada y puede ser creado utilizando la herramienta addpasswd.

- MinPoolSize=5
  Default: 1
  Define el número de conexiones activas SQL. Esto permite un mejor rendimiento, puesto que más de una consulta puede ser ejecutada al mismo tiempo. MinPoolSize=1 simula un viejo comportamiento, cuando el acceso a la base de datos es serializada (una consulta a la vez).

- RegQuery=SELECT ...
  Default: N/A
  Aquí se puede definir una consulta SQL y utilizarla para realizar autenticación y

autorización de registro de enpoints. La consulta es parametrizada, lo que quiere decir que se puede realizar reemplazo de parámetros antes de que cada consulta se ejecute. Se han definido los siguientes parámetros:

- %g - El identificador del gatekeeper.
- %{gkip} - IP del gatekeeper en el que se recibió la petición.
- %u - Username (nombre del usuario) asociado con un endpoint (usualmente un ID H.323)
- %{callerip} - IP del emisor (caller) (desde donde se ha recibido la petición - NAT IP para endpoints nateados)
- %{aliases} - Una lista de alias del endpoitn separados por una coma.

Si la consulta no devuelve filas (rows), el resultado no está definido, lo que básicamente significa "Fallo"(failure) para aquellas reglas de tipo required y "Pruebe la siguiente regla"para aquellas reglas de tipo optional. En el caso de que la consulta devuelva información, la primera fila (row) resultante es examinada para determinar el resultado de la autenticación y obtener información adicional:

1. Se convierte a la primera columna en un valor de tipo boolean (1, T, TRUE, allow, y, yes significan true) y es un resultado de autenticación (aceptar/rechazar).
2. Si el registro del endpoint se autentica satisfactoriamente, se examinan las columnas restantes:
   a) Si existe una columna llamada 'aliases', reemplazar el alias del endpoint original con los alias nuevos.
   b) Si existe una columna llamada 'billingmode', establezca un modo de billing asociado con el endpoint (0 - credit,
   c) 0 - debit)
   d) Si existe una columna llamada 'creditamount', establezca el balance de cuenta asociado con el endpoint (Ésta es una cadena arbritaria)

Algunas consultas de ejemplo:

```
SELECT 1, 0 AS billingmode, '12.00 USD' AS creditamount
SELECT NOT disabled, assignaliases AS aliases, balance FROM users WHERE h323id = '%u'
SELECT * FROM get_registration_auth('%g', '%u', '%{callerip}', '%{aliases}') AS result(accept, aliase
```

- NbQuery=SELECT ...
  Default: N/A
  Aqui se puede definir una consulta SQL para realizar la autenticación y autorización de las peticiones de ubicación (location requests) enviadas por los vecinos. La consulta es parametrizada, lo que quiere decir que se puede realizar el reemplazo de parámetros antes de que cada consulta se ejecute. Se han definido los siguientes parámetros:
  - %g - El identificador del gatekeeper.
  - %{gkip} - IP del gatekeeper en el que se recibió la petición.
  - %{nbid - El identificador del vecino.
  - %{nbip} - IP del vecino (desde donde se recibió la petición).
  - %{Calling-Station-Id} - Número del emisor (caller), si está disponible.
  - %{src-info} - Contenido del campo sourceInfo LRQ, si está disponible.

- %{Called-Station-Id} - Número de destino.
- %{dest-info} - Contenido del campo destinationInfo LRQ.
- %{bandwidth} - Ancho de banda pedido, si se encuentra en el LRQ.

Si la consulta no devuelve filas (rows), el resultado no está definido, lo que básicamente significa "Fallo"(failure) para aquellas reglas de tipo required y "Pruebe la siguiente regla"para aquellas reglas de tipo optional. En el caso de que la consulta devuelva información, la primera fila (row) resultante es examinada para determinar el resultado de la autenticación y obtener información adicional:

1. Se convierte a la primera columna en un valor de tipo boolean (1, T, TRUE, allow, y, yes significan true) y es un resultado de autenticación (aceptar/rechazar).
2. Si la pétición se autentica satisfactoriamente, se examinan las columnas restantes:
   a) Si existe una columna con el nombre 'destination', llenar el campo original destinationInfo con éstos alias nuevos - ésto afectará las desiciones de ruteo, la misma que se realiza después del paso .ªuth".

Alguna consultas de ejemplo:

```
SELECT active FROM neighbors WHERE name = '%{nbid}' AND ip = '%{nbip}' UNION SELECT 0
```

- CallQuery=SELECT ...
  Default: N/A
  Aqui se puede definir una consulta SQL para realizar autenticación y autorización de llamadas(ARQ y Setup). La consulta es parametrizada, lo que quiere decir que se puede realizar el reemplazo de parámetros antes de que cada consulta se ejecute. Se han definido los siguientes parámetros:
  - %g - El identificador del gatekeeper.
  - %{gkip} - IP del gatekeeper en el que se recibió la petición.
  - %u - Un nombre de usuario (username) asociado con el emisor (caller).
  - %{callerip} - IP del emisor (calle) (desde donde la petición ha sido recibida - NAT IP para endpoints nateados).
  - %{Calling-Station-Id} - Número del emisor (caller), si está disponible.
  - %{Called-Station-Id} - Número destino.
  - %{Dialed-Number} - Número de destino original (antes de la reescritura).
  - %{bandwidth} - Ancho de banda pedido, si está presente en los mensajes ARQ.
  - %{answer} - 1, Si la petición es una respuesta ARQ.
  - %{arq} - 1 para consulta ARQ activada, 0 para consulta Setup activada.

  Si la consulta no devuelve filas (rows), el resultado no está definido, lo que básicamente significa "Fallo"(failure) para aquellas reglas de tipo required y "Pruebe la siguiente regla"para aquellas reglas de tipo optional. En el caso de que la consulta devuelva información, la primera fila (row) resultante es examinada para determinar el resultado de la autenticación y obtener información adicional:

  1. Se convierte a la primera columna en un valor de tipo boolean (1, T, TRUE, allow, y, yes significan true) y es un resultado de autenticación (aceptar/rechazar).

2. Si el registro del endpoint se autentica satisfactoriamente, se examinan las columnas restantes:
   a) Si existe una columna con el nombre 'billingmode', establecer un modo de billing asociado con el endpoint (0 - credit,
   b) 0 - debit).
   c) Si existe una columna con el nombre 'creditamount', establecer el balance de cuenta asociado con el endpoint (Ésta es una cadena arbitraria).
   d) Si existe una columna con el nombre 'credittime', utilizar su valor entero para establecer el límite de duración de llamada.
   e) Si existe una columna con el nombre 'redirectnumber', reemplazar el número destino original con éste valor.
   f) Si existe una columna con el nombre 'redirectip', forzar a que la llamada sea enviada hacia una IP específica (aqui puede poner múltiples destinos separados por un punto y coma).
   g) Si existe una columna con el nombre 'proxy', forzar a que el gatekeeper habilite o deshabilite (dependiendo del valor de la columna 'proxy') el Proxy RTP para esta llamada.

Algunas consultas de ejemlo:

```
SELECT 1, 360 AS credittime, 0 AS proxy
SELECT * FROM auth_call('%g', '%u', '%{Calling-Station-Id}', '%{callerip}', '%{Called-Station-Id}') A
SELECT 1, '1234' AS redirectnumber, '192.168.1.1' AS redirectip
```

8.8. Sección [PrefixAuth]

La sección define la regla de autenticación para el módulo PrefixAuth. Actualmente, solamente ARQs y LRQs pueden ser autorizados por éste módulo.

Primero, se selecciona un prefijo muy específico de acuerdo al campo destinationInfo de la petición recibida. Entonces la petición es aceptada o rechazada de acuerdo al emparejamiento de reglas con la máscara de red más específica. Si no se emparejaron los prefijos y la opción default está especificada, la petición es aceptada o rechazada de acuerdo a eso. De lo contrario ésta es rechazada o pasada al siguiente módulo de autenticación de acuerdo a los requerimientos del módulo.

Formato:

```
prefix=authrule[|authrule|...]
```

Syntaxis:

```
<authrule> :=  <result> <authrule>

  <result>    := deny | allow
  <authrule>  := [!]ipv4:<iprule> | [!]alias:<aliasrule>
```

Donde <iprule> puede ser específicada en notación decimal punteada o en la notación CIDR, <aliasrule> está expresada en espreción regular. Si el signo '!' precede la regla, el sigificado es inverso.

Ejemplo:

```
555=deny ipv4:10.0.0.0/27|allow ipv4:0/0
5555=allow ipv4:192.168.1.1|deny ipv4:192.168.1.0/255.255.255.0
86=deny !ipv4:172.16.0.0/24
09=deny alias:^188884.*
ALL=allow ipv4:ALL
```

En esta configuración, todos los endpoints excepto los de la red 10.0.0.0/27 están permitidos llamar al prefijo 555 (excepto 5555). A los endpoints de la red 192.168.1.0/24 no se les permite llamar al prefijo 5555, excepto 192.168.1.1. Endpoints queno pertenecen a 172.16.0.0/24 no se les permite llamar al prefijo 86. Endpoints que tienen un alias que empieza con 188884 no se les permite llamar al prefijo 09. Todas las otras situaciones están permitidas.

8.9. Sección [RadAuth]

Esta sección define la configuración que posibilita la autenticación RADIUS basada en H.235 CATs (Cisco Access Tokens) presentes en las peticiones RAS: RRQ, ARQ y mensajes Q.931 Setup.

- Servers=SERVER1[:AUTH_PORT[:ACCT_PORT[:SECRET]]];SERVER2[:AUTH_PORT[:ACCT_PORT[:SECRET]]];...
  Default: N/A
  Aquí se definen los servidores RADIUS que van a ser utilizados para la autenticación. Esta lista puede contener un número arbitrario de servidores. El órden de los servidores es importante, porque los servidores serán interrogados por el módulo RADIUS en el órden dado. Si no se indica información del puerto, el número de puerto especificado en DefaultAuthPort será utilizado. Si tampoco se especifica el secret, el shared secret por defecto especificado en SharedSecret es utilizado. Los nombres de los servidores pueden ser direcciones IP o nombres DNS.

  Ejemplos de Servidores**:**

  ```
  Servers=192.168.1.1
  Servers=192.168.1.1:1645
  Servers=192.168.1.1:1645:1646:secret1
  Servers=radius1.mycompany.com:1812
  Servers=radius1.mycompany.com;radius2.mycompany.com
  Servers=radius1.mycompany.com:1812:1813:secret1;radius2.mycompany.com:1812:1813:secret2
  ```

- LocalInterface=IP_OR_FQDN
  Default: N/A
  Interfaz de red local particular que el cliente RADIUS debe utilizar para comunicarse con los servidores RADIUS. Este parámetro puede ser útil en máquinas NAT para restringir números de interfaces de red utilizados para la comunicación RADIUS. Por defecto este valor está vacío y permite a las peticiones RADIUS ser enviadas a cualquier (la que mejor conviene) interfaz de red. Si usted no está seguro de lo que está haciendo, es mejor dejar esta opción sin establecer.

- RadiusPortRange=10000-11000
  Default: N/A

  Por defecto (si esta opción no está establecida) el cliente RADIUS asigna puertos dinámicamente según lo especificado por el sistema operativo. Si usted desea

restringir para que un cliente RADIUS utilice solamente puertos de un rango particular, utilice este parámetro.

- DefaultAuthPort=PORT_NO
  Default: 1812

  Número de puerto por defecto que será utilizado para las peticiones de autenticacion RADIUS (Access-Request packets), si no se ha superpuesto por el atributo Servers.

- SharedSecret=SECRET
  Default: N/A (cadena vacía)

  Secret utilizada para autenticar este GnuGk (cliente NAS) con el servidor RADIUS. Éste debe ser un password criptográficamente fuerte. Éste es el valor por defecto utilizado, si no se colocó un secret específico en el parámetro Servers. Si EncryptAllPasswords está habilitada, o la variable KeyFilled está definida en esta sección, el password está en forma encriptada y debería ser creado utilizando la herramienta addpasswd.

- RequestTimeout=TIMEOUT_MS
  Default: 2000 (milisegundos)

  Tiempo en milisegundos que se debe esperar para que el servidor RADIUS responda a una petición enviada por el GnuGk. Si no se recibe ninguna respuesta dentro de este período de tiempo, se consultará al siguiente servidor RADIUS.

- IdCacheTimeout=TIMEOUT_MS
  Default: 9000 (milisegundos)
  Tiempo en milisegundos que se debe esperar para que identificadores de petición RADIUS de 8-bits sean únicos. Si el rango de identificadores de 8-bits se agota dentro de este período de tiempo un nuevo socket cliente (socket UDP) es asignado por el módulo RADIUS. Tomemos el ejemplo: Tenemos aproximadamente 60 RRQs/segundo; después de 4 segundos el rango de identificadores de 8-bits se agota; se asigna un nuevo socket; después de los siguientes 4 segundos, el segundo rango de identificadores de 8-bits se agota; un tercer socket es asignado; después del noveno segundo, identificadores del pool 1 estan disponibles otra vez - ... . En general, descanso demasiado largo significa demasiados recursos consumidos, descanso demasiado corto significa que el servidor RADIUS puede tomar los paquetes entrantes como duplicados y por lo tanto borrarlos.

- SocketDeleteTimeout=TIMEOUT_MS
  Default: 60000 (milisegundos) - 60 s
  Tiempo en milisegundos que se debe esperar para que aquellos sockets RADIUS no utilizados se cierren. Éste parámetro es utilizado conjuntamente con IdCacheTimeout - sockets adicionales creados durante los períodos pesados de carga del GK para atender peticiones entrantes. Estos sockets son cerrados durante los períodos ociosos.

- RequestRetransmissions=NUMBER
  Default: 2
  El número de veces en que una simple petición RADIUS es transmitida a cada servidor RADIUS configurado (si no se recibe respuesta). 1 (uno) significa no retransmisión,

2 (dos) significa retransmisión simple, ... El método exácto de retransmisión se define en el atributo RoundRobinServers.

- RoundRobinServers=BOOLEAN
  Default: 1
  Método de retransmisión de peticiones RADIUS.

  Si se fija en 1 (uno), la petición RADIUS es transmitida de la siguiente manera (Hasta que se recibe respuesta):

  ```
  Server #1 Attempt #1, Server #2 Attempt #1, ..., Server #N Attempt #1
  ...
  Server #1 Attempt #RequestRetransmissions, ..., Server #1 Attempt #RequestRetransmissions
  ```

  Si se fija en 0 (cero), se mantiene la siguiente secuencia:

  ```
  Server #1 Attempt #1, ..., Server #1 Attempt #RequestRetransmissions
  ...
  Server #N Attempt #1, ..., Server #N Attempt #RequestRetransmissions
  ```

- AppendCiscoAttributes=BOOLEAN
  Default: 0
  En el caso de que se utilice este parámetro, se debe incluir un atributo Cisco Vendor Specific RADIUS en la petición RADIUS (h323-conf-id,h323-call-origin,h323-call-type).

- IncludeTerminalAliases=BOOLEAN
  Default: 1
  En el caso de que se utilice, se debe enviar un atributo Cisco VSA 'h323-ivr-out' con una lista de alias con las que el endpoint se está registrando (RRQ.m_terminalAlias). Este atributo se utiliza para proveer un control detallado sobre la lista de alias con los cuales el endpoint es pemitido registrarse. El formato de este atributo es el siguiente:

  ```
          Cisco-AV-Pair = "h323-ivr-out=terminal-alias:" alias [,alias] [;]
  Example:
          Cisco-AV-Pair = "h323-ivr-out=terminal-alias:helpdesk,support,77771;"
  ```

- UseDialedNumber=BOOLEAN
  Default: 0
  Seleccione el tipo de número Called-Station-Id entre el original (según lo marcado por el usuario) - UseDialedNumber=1 - y el reescrito - UseDialedNumber=0.

8.10. Sección [RadAliasAuth]

En esta sección se definen parámetros de configuración que permiten la autenticación RADIUS basada en alias de endpoints y/o direcciones IP presentes en las peticiones RAS: RRQ o en las peticiones Setup: Q.931. Este esquema de autenticación es muy utilizado tanto para endpoints registrados en el gatekeeper (ARQ,RRQ) como para llamadas que vienen desde endpoints no registrados(Setup).

- Servers=SERVER1[:AUTH_PORT[:ACCT_PORT[:SECRET]]];SERVER2[:AUTH_PORT[:ACCT_PORT[:SECRET]]];...
  Default: N/A
  Servidores RADIUS que van a ser utilizados para autenticación de peticiones RAS. Esta lista puede contener un número arbitrario de servidores. El órden de los

servidores es importante, porque los servidores serán interrogados por el módulo RADIUS en el órden dado. Si no se indica información del puerto, el número de puerto especificado en DefaultAuthPort será utilizado. Si tampoco se especifica el secret, el shared secret por defecto especificado en SharedSecret es utilizado. Los nombres de los servidores pueden ser direcciones IP o nombres DNS.

Example:

```
Servers=192.168.3.1:1645;192.168.3.2:1812:1813:mysecret;radius.mycompany.com
```

- LocalInterface=IP_OR_FQDN
  Default: N/A
  Interfaz de red local particular que el cliente RADIUS debe utilizar para comunicarse con los servidores RADIUS. Este parámetro puede ser útil en máquinas NAT para restringir números de interfaces de red utilizados para la comunicación RADIUS. Por defecto este valor está vacío y permite a las peticiones RADIUS ser enviadas a cualquier (la que mejor conviene) interfaz de red. Si usted no está seguro de lo que está haciendo, es mejor dejar esta opción sin establecer.
- RadiusPortRange=10000-11000
  Default: N/A
  Por defecto (si esta opción no está establecida) el cliente RADIUS asigna puertos dinámicamente según lo especificado por el sistema operativo. Si usted desea restringir a cliente RADIUS a utilizar solamente puertos de un rango particular, utilice este parámetro.
- DefaultAuthPort=PORT_NO
  Default: 1812
  Número de puerto por defecto que será utilizado para las peticiones de autenticacion RADIUS (Access-Request packets), si no se ha suerpuesto por el parámetro Servers.
- SharedSecret=SECRET
  Default: N/A (cadena vacía) Secret utilizada para autenticar éste GnuGk (cliente NAS) con el servidor RADIUS. Este debe ser un password criptográficamente fuerte. Este es el valor por defecto utilizado, si no se colocó un secret específico en el parámetro Servers. Si EncryptAllPasswords está habilitada, o la variable KeyFilled está definida en esta sección, el password está en forma encriptada y debería ser creado utilizando la herramienta addpasswd.
- RequestTimeout=TIMEOUT_MS
  Default: 2000 (milisegundos)
  Tiempo en milisegundos para que el servidor RADIUS responda a una petición enviada por el GnuGk. Si no se recibe ninguna respuesta dentro de este período de tiempo, se consultará al siguiente servidor RADIUS.
- IdCacheTimeout=TIMEOUT_MS
  Default: 9000 (milisegundos)
  Tiempo en milisegundos para que identificadores de petición RADIUS de 8-bits sean únicos. Si el rango de identificadores de 8-bits se agota dentro de este período de tiempo un nuevo socket cliente (socket UDP) es asignado por el módulo RADIUS. Tomemos el ejemplo: Tenemos aproximadamente 60 RRQs/segundo; después de 4 segundos el rango de identificadores de 8-bits se agota; se asigna un nuevo socket; despúés de los siguientes 4 segundos, el segundo rango de identificadores de 8-bits se agota; un tercer socket es asignado; después del noveno segundo, identificadores

del pool 1 estan disponibles otra vez - ... . En general, descanso demasiado largo significa demasiados recursos consumidos, descanso demasiado corto significa que el servidor RADIUS puede tomar los paquetes entrantes como duplicados y por lo tanto borrarlos.

- SocketDeleteTimeout=TIMEOUT_MS
  Default: 60000 (miliseconds) - 60 s
  Tiempo en milisegundos para que sockets RADIUS no utilizados sean cerrados. Éste parámetro es utilizado conjuntamente con IdCacheTimeout - sockets adicionales creados durante los períodos pesados de carga del GK para atender peticiones entrantes. Estos sockets se cierran durante los períodos ociosos.

- RequestRetransmissions=NUMBER
  Default: 2
  El número de veces en que una simple petición RADIUS es transmitida a cada servidor RADIUS configurado (si no se recibe respuesta). 1 (uno) significa no retransmisión, 2 (dos) significa retransmisión simple, ... El método exácto de retransmisión es definido por el atributo RoundRobinServers.

- RoundRobinServers=BOOLEAN
  Default: 1
  Método de retransmisión de peticiones RADIUS.

  Si se fija en 1 (uno), la petición RADIUS es transmitida de la siguiente manera (Hasta que se recibe respuesta):

  ```
  Server #1 Attempt #1, Server #2 Attempt #1, ..., Server #N Attempt #1
  ...
  Server #1 Attempt #RequestRetransmissions, ..., Server #1 Attempt #RequestRetransmissions
  ```

  Si se fija en 0 (cero), se mantiene la siguiente secuencia:

  ```
  Server #1 Attempt #1, ..., Server #1 Attempt #RequestRetransmissions
  ...
  Server #N Attempt #1, ..., Server #N Attempt #RequestRetransmissions
  ```

- AppendCiscoAttributes=BOOLEAN
  Default: 1
  En el caso de que se utilice este parámetro, se debe incluir un atributo Cisco Vendor Specific RADIUS en la petición RADIUS (h323-conf-id,h323-call-origin,h323-call-type).

- IncludeTerminalAliases=BOOLEAN
  Default: 1 En el caso de que se utilice este parámetro, se debe enviar un atributo Cisco VSA 'h323-ivr-out' con una lista de alias con los que el endpoint se está registrando (RRQ.m_terminalAlias). Este atributo se utiliza para proveer un control detallado sobre la lista de alias con los cuales el endpoint es pemitido registrarse. El formato de este atributo es el siguiente:

  ```
          Cisco-AV-Pair = "h323-ivr-out=terminal-alias:" alias [,alias] [;]
  Example:
          Cisco-AV-Pair = "h323-ivr-out=terminal-alias:helpdesk,support,77771;"
  ```

- FixedUsername
  Default: N/A
  Si se establece este parámetro, éste se superpone sobre el valor del atributo

User-Name RADIUS para peticiones RADIUS salientes. Esto significa que cada petición de acceso (Access-Request) se autenticará en cuanto al usuario FixedUsername.

- FixedPassword
  Default: N/A
  Si no esta activada esta opción, User-Password es una copia de User-Name. Por ejemplo, si User-Name es 'john' entonces el valor de User-Password será también 'john'. Activar este parámetro sustituye este comportamiento y el atributo User-Password será siempre establecido al valor de FixedPassword. Si EncryptAllPasswords está habilitada, o la variable KeyFilled está definida en esta sección, el password está en forma encriptada y debe ser creada utilizando la herramienta addpasswd.

  Ejemplo 1:

  ```
  (Neither FixedUsername nor FixedPassword set)
  ```

  Todos los endpoints se autenticarán utilizando sus alias como username y el password. Esto significa, por ejemplo que el endpoint 'EP1' se autenticará con el username 'EP1' y el password 'EP1'.

  Ejemplo 2:

  ```
  (FixedUsername not set)
  FixedPassword=ppp
  ```

  Todos los endpoints se autenticarán utilizando como alias y passwords la palabra 'ppp'.

  Ejemplo 3:

  ```
  FixedUsername=ppp
  FixedPassword=ppp
  ```

  Todos los endpoints se autenticarán utilizando como username la palabra 'ppp' y como password la palabra 'ppp'.

- UseDialedNumber=BOOLEAN
  Default: 0
  Seleccione el número Called-Station-Id entre el original (como marcado por el usuario) - UseDialedNumber=1 - y el y el reescrito - UseDialedNumber=0.

8.11. Sección [CapacityControl]

Esta sección contiene un conjunto de reglas que permiten controlar el volumen de llamadas entrantes dependiendo de varias condiciones. Para que este módulo funcione, los módulos de autenticación y de accounting CapacityControl deben estar habilitados como se muestra acontinuación:

```
[Gatekeeper::Auth]
CapacityControl=required;Setup

[Gatekeeper::Acct]
CapacityControl=required;start,stop
```

Una regla de capacidad (capacity rule) puede corresponderse con una IP de emisor (caller), ID H.323 del emisor (caller) y/o un número CLI de emisor (caller) - en el órden especificado. Adicionalmente, la correspondencia puede ser reducida especificando un patrón de número llamado (called number). Este módulo funciona manteniendo listas de llamadas actuales para cada ruta entrante (regla) - ésto se consigue teniendo configurado el módulo de accounting CapacityControl para que agregue o elimine llamadas activas de las rutas que coincidan. El módulo de autenticación CapacityControl revisa las reglas y acepta o rechaza una llamada tomando en cuenta el volumen de llamadas actual o máximo para una ruta entrante que coincida.

Formato para una regla de ruta entrante:

[ip:CALLER_IP|h323id:CALLER_H323ID|cli:CALLER_NUMBER]=[CALLED NUMBER REGEX PATTERN] MAX_CAPACITY

ip:, h323id: y cli: son prefijos que definen el tipo de regla. Una llamada entrante será emparejada con la IP del emisor (caller), H.323ID o por el CLI. El valor opcional CALLED NUMBER REGEX PATTERN es una expresión regular que el número llamado (called number) debe emparejar para aplicar esta regla. MAX_CAPACITY Es un número máximo de llamadas activas para esta ruta.

Ejemplo 1:

```
[CapacityControl]
ip:192.168.1.0/24=30
ip:any=120
```

Estas reglas indican que la subred 192.168.1.0/24 puede enviar hasta 30 llamadas concurrentes, mientras que otras IPs pueden enviar hasta 120 llamadas concurrentes.

Ejemplo 2:

```
[RewriteCLI]
%r1% cli:1001=30
%r2% cli:1001=^48(50|51) 5
```

Estas reglas limitan al emisor (caller) con un CLI de 1001 a realizar hasta 5 llamadas hacia los destinos 4850/4851 y hasta 30 llamadas hacia otros destinos. %r1% y %r2% son constructores especiales que permiten tener la misma clave de configuración cli:1001 mas de una vez.

9. Configuración de Accounting

Accounting se denomina a las operaciones que los usuarios realizarán en el gatekeeper: inicio de una llamada, detención y retroalimentación de la misma; e incluso el inicio y detención del propio gatekeeper.

Las secciones siguientes en el Archivo de Configuración pueden ser utilzadas para configurar el Accounting.

9.1. Sección [Gatekeeper::Acct]

Esta sección define una lista de módulos que serán los que realicen el accounting. El accounting es para registrar eventos de encendido/apagado del gatekeeper y eventos de

inicio/detención/actualización de llamada. Cada módulo de accounting registra eventos recibidos a un módulo de almacenamiento específico. Dicho almacenamiento puede ser en texto plano o un servidor RADIUS y muchos más. La configuración es muy similar a la de autenticación de gatekeeper. Puede revisar la sección 8.1 ([Gatekeeper::Auth]) para más detalle.

Todos los CDRs también serán enviados al puerto de estado y podrán ser utilizados por aplicaciones externas.

Sintaxis:

```
acctmod=acciones

 <acctmod>  := FileAcct | RadAcct | SQLAcct | StatusAcct | SyslogAcct | CapacityControl | ...
 <acciones> := <control>[;<evento>,<evento>,...]
 <control>  := optional | required | sufficient | alternative
 <evento>   := start | stop | connect | update | on | off
```

La lista de eventos le indica al gatekeeper cuales eventos debe capturar o registrar con el módulo de accounting dado (si un tipo de evento es soportado por el módulo):

- start - Una llamada ha sido iniciada y un mensaje de Configuración ha sido recibido (solo disponible en modo enrutado),
- connect - Una llamada ha sido establecida (solo disponible en modo enrutado),
- update - Una llamada está activa y se realizó la actualización periódica para reflejar la duración de la nueva llamada. La Frecuencia de estas actualizaciones está determinada por el parámetro AcctUpdateInterval definido en la sección 12.1 ([CallTable]),
- stop - Una llamada ha sido desconectada o parada (eliminada de la tabla de llamadas de GK),
- on - El gatekeeper ha sido iniciado,
- off - El gatekeeper ha sido apagado.

Un evento registrado por un módulo puede resultar en uno de estos tres códigos: ok, fail, next.

- ok - El evento ha sido registrado correctamente por este módulo,
- fail - El módulo falló al registrar el evento,
- next - El evento no ha sido registrado por este módulo, porque el módulo no está configurado para o no soporta este tipo de evento.

Los módulos de accounting pueden ser apilados para registrar eventos de multiples módulos o para crear configuraciones a prueba de fallos. La bandera control para cada módulo, junto con los códigos de resultado, definen lo que es el estado final del evento procesado por la pila completa de módulos. Si el resultado final es failure, pueden tener lugar algunas acciones especiales. Actualmente, si la anotación de un evento de inicio de llamada (start) falla, la llamada es desconectada inmediatamente. Los siguientes banderas control están reconocidas:

- required - Si el módulo falla al aregistrar un evento, el estado final es establecido a "Falloz el evento es pasado a cualquiera de los módulos restantes,
- optional - El módulo intenta registrar un evento, pero el estado final no es afectado por el éxito o fracaso (a excepción de cuando el módulo es el último en la lista). El evento es siempre pasado a cualquiera de los módulos restantes,
- sufficient - El módulo determina el estado final. Si un evento es registrado satisfactoriamente, ningún módulo más es procesado. En caso contrario el estado final es establecido como un "Falloz el evento es pasado a cualquiera del resto de módulos,
- alternative - Si el módulo registra un evento satisfactoriamente, ningún módulo más será procesado. En caso contrario el estado final no será modificado y el evento es pasado a cualquiera de los módulos restantes.

Módulos de acconting actualmente soportados:

- FileAcct Un archivo de texto plano es el que registra los CDRs. Éste módulo escribe una línea de estado como una línea CDR en un archivo de texto especifico. Este modulo soporta solo el evento de accounting stop. Las opciones de configuración serán revisadas en la sección 9.2 ([FileAcct]).
- RadAcct Este módulo realiza accounting a través de RADIUS. Éste módulo soporta todos los tipos de eventos (start, stop, update, on, off). Revisar la sección 9.3 ([RadAcct]) para detalles de configuración.
- SQLAcct Este módulo realiza accounting a través de SQL. Éste módulo soporta los tipos de evento (start, connect, stop, update). Revisar la sección 9.4 ([SQLAcct]) para detalles de configuración.
- StatusAcct Este módulo registra todos los eventos de accounting en el puerto de estado. Puede ser utilizado como interfaz hacia aplicaciones externas en tiempo real. Este módulo soporta los tipos de evento (start, connect, stop, update). Revisar la sección 9.5 ([StatusAcct]) para detalles de configuración.
- SyslogAcct Este módulo registra todos los eventos de accounting en el registro del sistema Unix (syslog) Unix. Este módulo soporta los eventos (start, connect, stop, update). Revisar la sección 9.6 ([SyslogAcct]) para detalles de configuración.
- CapacityControl Este módulo realiza registro de volúmen de llamadas entrantes, es indispensable para que el módulo de autenticación CapacityControl funcione correctamente. Revisar la sección 8.11 ([CapacityControl]) para mayor detalle.
- default Este es un pseudo-módulo especial y es utilizado para configurar el estado final si los módulos precedentes no lo han determinado. El formato es como sigue:

  Sintaxis:

```
default=<estado>[;<evento>,<evento>,...]
<estado> := accept | fail
<evento> := start | stop | update | on | off
```

El ejemplo de configuración 1 intenta registrar el inicio/fin de una llamada con un servidor RADIUS, y siempre escribe un CDR en un archivo de texto:

Ejemplo:

```
RadAcct=optional;start,stop
FileAcct=required
```

El ejemplo de configuración 2 intenta registrar el inicio/fin de una llamada con un servidor RADIUS, si ésto falla utiliza un archivo para registrar los CDRs:

Ejemplo:

```
RadAcct=alternative;start,stop
FileAcct=sufficient;stop
default=accept
```

La regla default es requerida aquí para prevenir que la llamada sea rechazada por si la anotación del evento de inicio de RadAcct falla. Si RadAcct retorna el código fail, se pasa al módulo FileAcct. El módulo FileAcct no soporta el evento start, así que retorna el código next. Si no estuviese la regla default, el estado final sería failure, porque ningún módulo ha sido capaz de anotar el evento.

El ejemplo de configuración 3 siempre anota los eventos de inicio y fin de llamada con un servidor RADIUS, si este falla para el evento de fin de llamada, utiliza un archivo CDR para almacenar la información de la llamada:

Ejemplo:

```
RadAcct=alternative;start,stop
FileAcct=sufficient;stop
default=fail;start
```

La regla default es opcional aquí. Si RadAcct retorna el código fail para el evento start, el código es pasado al módulo FileAcct. El módulo FileAcct no soporta el evento start, así que retorna el código next. La regla default se asegura de que la llamada está desconectada si el evento de inicio de llamada puede no haber sido anotado con RadAcct. Pero deseamos almacenar un CDR en un fichero de texto en caso de que el servidor RADIUS esté caído cuando la llamada se desconecte, así podemos extraer la duración de la llamada dentro del sistema de facturación más tarde.

9.2. Sección [FileAcct]

Este módulo de accounting escribe líneas CDR en un archivo de texto específico. El formato CDR puede ser uno estándar (el mismo que se puede ver por la interfaz de estado) o uno personalizado (usando una cadena de consulta parametrizada).

- DetailFile=RUTA_COMPLETA_Y_NOMBRE_DE_ARCHIVO
  Default: N/A
  Defina la ruta completa para el fichero de texto plano CDR. Si un fichero con el nombre dado ya existe, los nuevos CDRs serán añadidos al final del fichero.
- StandardCDRFormat=0
  Default: 1

Utilice un formato CDR compatible con el formato de CDR de la interfaz de estado (1) o cree cadenas CDR personalizadas desde la cadena parametrizada CDRString.

- CDRString=%s|%g|%u|%{Calling-Station-Id}|%{Called-Station-Id}|%d|%c
  Default: N/A
  Si StandardCDRFormat está desactivado (0) o no se especifica nada, esta cadena parametrizada le indica al gatekeeper cómo crear CDRs personalizados. Los parámetros son especificados usando el caracter % y pueden ser una letra (como %n) o más largos (como %{CallId}). Cualquier caracter restante que no sea nombre de parámetro será simplemente copiado al final de la cadena CDR. Los parámetros reconocidos son los siguientes:
  - %g - Nombre del gatekeeper.
  - %n - Número de llamada (no es único después del reinicio del gatekeeper).
  - %d - Duración de la llamada (segundos).
  - %t - Duración total de la llamada (desde Setup hasta Release Complete).
  - %c - Causa de desconexión Q.931 (entero decimal).
  - %r - Quién desconecto la llamada (-1 - desconocido, 0 - el gatekeeper, 1 - el emisor, 2 - el receptor).
  - %p - PDD (Post Dial Delay) en segundos.
  - %s - Identificador de sesión, único (para este gatekeeper) (Acct-Session-Id).
  - %u - H.323 ID de la parte llamada.
  - %{gkip} - Dirección IP address del gatekeeper.
  - %{CallId} - Identificador H.323 de la llamada (16 dígitos hex de 8-bits).
  - %{ConfId} - Identificador H.323 de la conferencia (16 dígitos hex de 8-bits).
  - %{setup-time} - Cadena de hora y fecha para el mensaje de Setup Q.931.
  - %{alerting-time} - Cadena de hora y fecha para el mensaje de Alerting Q.931.
  - %{connect-time} - Cadena de hora y fecha para un evento de llamada conectada.
  - %{disconnect-time} - Cadena de hora y fecha para un evento de llamada desconectada.
  - %{ring-time} - Tiempo que un teléfono remoto estuvo sonando (desde Alerting hasta Connect o Release Complete).
  - %{caller-ip} - Dirección IP del emisor.
  - %{caller-port} - Puerto de señalización por el que llama el emisor.
  - %{callee-ip} - Dirección IP del receptor
  - %{callee-port} - Puerto de señalización por el que recibe la llamada el receptor.
  - %{src-info} - Una lista separada por comas de alias del emisor.
  - %{dest-info} - Una lista separada por comas de alias del receptor.
  - %{Calling-Station-Id} - Número del emisor
  - %{Called-Station-Id} - Número del receptor (reescrito)
  - %{Dialed-Number} - Número marcado (como es recibido por la parte llamada).
  - %{caller-epid} - Identificador del endpoint del emisor.
  - %{callee-epid} - Identificador del endpoint del receptor.

- %{call-attempts} - Número de intentos para establecer la llamada (con failover esta puede ser > 1).
- %{last-cdr} - Éste es el último CDR para esta llamada ? (0 / 1) solamente cuando utilice failover éste puede ser 0.
- %{media-oip} - Caller's RTP media IP (solamente para llamadas ruteadas o dirigidas con H.245).
- %{codec} - Codec de audio utilizado durante la llamada (solamente para llamadas ruteadas o dirigidas H.245).

- TimestampFormat=Cisco
  Default: N/A
  Formato de las cadenas de fecha y hora impresas en las cadenas CDR. Si esta configuración no se especifica, se aplicará una configuración global de la sección principal del gatekeeper.

- Rotate=hourly | daily | weekly | monthly | L... | S...
  Default: N/A
  Si se configura, el archivo CDR será rotado basado en esta configuración. La rotación Hourly habilita una rotación por hora, la daily - una por día, weekly - una por semana y monthly - una por mes. El momento exacto de la rotación es determinado por una combinación de RotateDay y RotateTime. Durante la rotación, un fichero existente será renombrado a CURRENT_FILENAME.YYYYMMDD-HHMMSS, donde YYYYMMDD-HHMMSS es reemplazado con la hora y fecha actuales y nuevos CDRs son registrados a un archivo vacío.
  Además, la rotación por número de CDRs escritos (L...) y por tamaño de fichero (S...) están soportadas. El prefijo L especifica un número de líneas CDR escritas, el prefijo S especifica el tamaño de un fichero CDR. Los sufijos k y m pueden ser utilizados para especificar miles (kilobytes) y millones (megabytes) de bytes. Ver los ejemplos para más detalles.

Ejemplo 1 - sin rotación:

```
[FileAcct]
DetailFile=/var/log/gk/cdr.log
```

Ejemplo 2 - rotación cada hora (00:45, 01:45, ..., 23:45):

```
[FileAcct]
DetailFile=/var/log/gk/cdr.log
Rotate=hourly
RotateTime=45
```

Ejemplo 3 - rota cada día a las 23:00 (11PM):

```
[FileAcct]
DetailFile=/var/log/gk/cdr.log
Rotate=daily
RotateTime=23:00
```

Ejemplo 4 - rota cada Domingo a las 00:59:

```
[FileAcct]
DetailFile=/var/log/gk/cdr.log
```

```
Rotate=weekly
RotateDay=Sun
RotateTime=00:59
```

Ejemplo 5 - rota el último día de cada mes:

```
[FileAcct]
DetailFile=/var/log/gk/cdr.log
Rotate=monthly
RotateDay=31
RotateTime=23:00
```

Ejemplo 6 - rota cada 10000 CDRs:

```
[FileAcct]
DetailFile=/var/log/gk/cdr.log
Rotate=L10000
```

Ejemplo 7 - rota cada 10 kilobytes:

```
[FileAcct]
DetailFile=/var/log/gk/cdr.log
Rotate=S10k
```

9.3. Sección [RadAcct]

Este módulo de accounting envía datos de accounting a un servidor RADIUS. La configuración del módulo es casi lo mismo que para los autenticadores RADIUS (ver 8.9 ([RadAuth]) y 8.10 ([RadAliasAuth]) para mayor detalles sobre los parámetros).

- Servers=SERVER1[:AUTH_PORT:ACCT_PORT[:SECRET]];SERVER2[:AUTH_PORT:ACCT_PORT[:SECRET]];...
  Default: N/A
  Servidores RADIUS a los que se enviará los datos de accounting. Si no se especifica información sobre el puerto, el puerto utilizado se tomará del parámetro DefaultAcctPort. Si SECRET no está configurado, por defecto se usará el secret compartido de SharedSecret. El nombre del servidor puede ser tanto una dirección IP como un nombre DNS.

  Líneas de Servidores de ejemplo:

  ```
  Servers=192.168.1.1
  Servers=192.168.1.1:1645:1646
  Servers=192.168.1.1:1645:1646:secret1
  Servers=radius1.mycompany.com:1812:1813
  Servers=radius1.mycompany.com;radius2.mycompany.com
  Servers=radius1.mycompany.com:1812:1813:secret1;radius2.mycompany.com:1812:1813:secret2
  ```

- LocalInterface=IP_OR_FQDN
  Default: N/A
  Interfaz de red local en particular que el cliente RADIUS deberá usar para comunicarse con los servidores RADIUS.

- RadiusPortRange=10000-11000
  Default: N/A
  Por defecto (si esta opción no está establecida) el cliente RADIUS asigna los puertos dinámicamente como esté especificado por el sistema operativo. Si usted desea restringir al cliente RADIUS para que utilice puertos solamente desde un rango particular - active este parámetro.

- DefaultAcctPort=PORT_NO
  Default: 1813
  El puerto por defecto que será utilizado para las peticiones de accounting de RADIUS, si no se superpone por el atributo Servers.

- SharedSecret=SECRET
  Default: N/A (cadena vacía)
  Un "secretütilizado para autenticar este GnuGK (cliente NAS) con el servidor RADIUS. Ésta debe ser una contraseña criptográficamente sólida. Éste es el valor utilizado por defecto, si no se especifica secret en Servers. Si EncryptAllPasswords está activado, o la variable KeyFilled está definida en esta sección, la contraseña está en forma encriptada y debe ser creada usando la herramienta addpasswd.

- RequestTimeout=TIMEOUT_MS
  Default: 2000 (milisegundos)
  Tiempo de espera (milisegundos) para que un servidor RADIUS responda a una solicitud enviada por GnuGK. Si no se recibe una respuesta dentro de este periodo de tiempo, se consultará al siguiente servidor RADIUS.

- IdCacheTimeout=TIMEOUT_MS
  Default: 9000 (milisegundos)
  Tiempo de espera (milisegundos) para que identificadores de petición RADIUS de 8-bits sean únicos.

- SocketDeleteTimeout=TIMEOUT_MS
  Default: 60000 (milisegundos) - 60 s
  Tiempo que se espera para cerrar sockets RADIUS no utilizados.

- RequestRetransmissions=NUMBER
  Default: 2
  Cuantas veces una simple solicitud RADIUS es transmitida a cada servidor RADIUS configurado (si no se recibe una respuesta).

- RoundRobinServers=BOOLEAN
  Default: 1
  Método de retransmisión de solicitudes RADIUS.

- AppendCiscoAttributes=BOOLEAN
  Default: 0
  Si está activa, los atributos RADIUS específicos de fabricante Cisco son incluídos en las solicitudes RADIUS (h323-conf-id,h323-call-origin,h323-call-type).

- TimestampFormat=ISO8601
  Default: N/A
  Formato de las cadenas de fecha y hora enviadas en los atributos RADIUS. Si este atributo no está especificado, uno global de la sección principal de gatekeeper será aplicado.

- UseDialedNumber=BOOLEAN
  Default: 0
  Selecciona el tipo de número Called-Station-Id entre el original (como fue marcado por el usuario) - UseDialedNumer=1 - y el reescrito - UseDialedNumer=0.

9.4. Sección [SQLAcct]

Este módulo de accounting almacena información de accounting directamente a una base de datos SQL. Muchas opciones de configuración son comunes con otros módulos SQL.

- Driver=MySQL | PostgreSQL | Firebird
  Default: N/A
  El driver de base de datos SQL a usar. Actualmente, los manejadores implementados son MySQL, PostgreSQL y Firebird.
- Host=DNS[:PORT] | IP[:PORT]
  Default: localhost
  Dirección de la máquina del servidor SQL. Puede estar en forma de DNS[:PORT] o IP[:PORT]. Como sql.mycompany.com o sql.mycompany.com:3306 o 192.168.3.100.
- Database=billing
  Default: billing
  El nombre de la base de datos a la cual se realiza la conexión.
- Username=gnugk
  El nombre de usuario utilizado para conectarse a la base de datos.
- Password=secret
  La contraseña utilizada para conectarse a la base de datos. Si la contraseña no se especifica, se intentará una conexión a la base de datos sin ninguna contraseña. Si EncryptAllPasswords está activo, o una variable KeyFilled es definida en esta sección, la contraseña está en una forma encriptada y debe ser creada usando la herramienta addpasswd.
- StartQuery=INSERT ...
  Default: N/A
  Define una consulta SQL, utilizada para insertar un nuevo registro de llamada a la base de datos. La consulta es parametrizada - eso significa que el reemplazo de parámetros se hace antes de que cada consulta sea ejecutada. Los parámetros son precedidos por el caracter % y pueden ser una letra (como %u) o cadenas enteras (como %{src-info}). Especificar %% para añadir un caracter de porcentaje dentro de la cadena de consulta (como %%). Para SQLAcct se pueden utilizar los siguientes parámetros:
  - %g - Nombre del gatekeeper.
  - %n - Número de llamada (no es único después del reinicio del gatekeeper).
  - %d - Duración de la llamada (segundos).
  - %t - Duración total de la llamada (desde Setup hasta Release Complete).
  - %c - Causa de desconexión Q.931 (entero decimal).
  - %r - Quién desconecto la llamada (-1 - desconocido, 0 - el gatekeeper, 1 - el emisor, 2 - el receptor).

- %p - PDD (Post Dial Delay) en segundos.
- %s - Identificador de sesión, único (para este gatekeeper) (Acct-Session-Id).
- %u - H.323 ID de la parte llamada.
- %{gkip} - Dirección IP address del gatekeeper.
- %{CallId} - Identificador H.323 de la llamada (16 dígitos hex de 8-bits).
- %{ConfId} - Identificador H.323 de la conferencia (16 dígitos hex de 8-bits).
- %{setup-time} - Cadena de hora y fecha para el mensaje de Setup Q.931.
- %{alerting-time} - Cadena de hora y fecha para el mensaje de Alerting Q.931.
- %{connect-time} - Cadena de hora y fecha para un evento de llamada conectada.
- %{disconnect-time} - Cadena de hora y fecha para un evento de llamada desconectada.
- %{ring-time} - Tiempo que un teléfono remoto estuvo sonando (desde Alerting hasta Connect o Release Complete).
- %{caller-ip} - Dirección IP del emisor.
- %{caller-port} - Puerto de señalización por el que llama el emisor.
- %{callee-ip} - Dirección IP del receptor
- %{callee-port} - Puerto de señalización por el que recibe la llamada el receptor.
- %{src-info} - Una lista separada por comas de alias del emisor.
- %{dest-info} - Una lista separada por comas de alias del receptor.
- %{Calling-Station-Id} - Número del emisor
- %{Called-Station-Id} - Número del receptor (reescrito)
- %{Dialed-Number} - Número marcado (como es recibido por la parte llamada).
- %{caller-epid} - Identificador del endpoint del emisor.
- %{callee-epid} - Identificador del endpoint del receptor.
- %{call-attempts} - Número de intentos para establecer la llamada (con failover esta puede ser > 1).
- %{last-cdr} - Éste es el último CDR para esta llamada ? (0 / 1) solamente cuando utilice failover éste puede ser 0.
- %{media-oip} - Caller's RTP media IP (solamente para llamadas ruteadas o dirigidas con H.245).
- %{codec} - Codec de audio utilizado durante la llamada (solamente para llamadas ruteadas o dirigidas H.245).

Consulta de ejemplo:

```
INSERT INTO call (gkname, sessid, username, calling, called)
       VALUES ('%g', '%s', '%u', '%{Calling-Station-Id}', '%{Called-Station-Id}')
```

- StartQueryAlt=INSERT ...
  Default: N/A
  Define una consulta SQL utilizada para insertar un nuevo registro de llamada en la base de datos en caso de que StartQuery falle por alguna razón (la llamada ya existe, por ejemplo). La sintaxis y parámetros son los mismos que para StartQuery.

- UpdateQuery=UPDATE ...
  Default: N/A
  Define una consulta SQL utilizada para actualizar un registro de llamada en una base de datos con el estado actual de llamada. La sintaxis y parámetros son los mismos que para StartQuery.

  Ejemplo de consulta:

  ```
  UPDATE call SET duration = %d WHERE gkname = '%g' AND sessid = '%s'
  ```

- StopQuery=UPDATE ...
  Default: N/A
  Define una consulta SQL utilizada para actualizar el registro de llamada en una base de datos cuando la llamada ha finalizado (disconnected). La sintaxis y parámetros son los mismos que para StartQuery.

  Ejemplo de consulta:

  ```
  UPDATE call SET duration = %d, dtime = '%{disconnect-time}' WHERE gkname = '%g' AND sessid = '%s'
  ```

- StopQueryAlt=INSERT ...
  Default: N/A
  Define una consulta SQL utilizada para actualizar un registro de llamada en una base de datos cuando la llamada ha finalizado (disconnected) en caso de que StopQuery falle (porque el registro de llamada no exista ya, por ejemplo). La sintaxis y parámetros son los mismos que para StartQuery.

  Ejemplo de consulta:

  ```
  INSERT INTO call (gkname, sessid, username, calling, called, duration)
         VALUES ('%g', '%s', '%u', '%{Calling-Station-Id}', '%{Called-Station-Id}', %d)
  ```

- TimestampFormat=MySQL
  Default: N/A
  Formato de cadena de fecha y hora utilizada en las consultas. Si este atributo no es especificado, La confifuración global de la sección principal de gatekeeper será aplicada.

- MinPoolSize=5
  Default: 1
  Número de conecciones SQL concurrentes dentro del pool de conexiones. La primera conexión disponible dentro del pool es utilizada pata almacenar datos de accounting.

9.5. Sección [StatusAcct]

Este módulo de accounting envía toda la información del proceso de accounting hacia el puerto de estado donde podrá ser utilizada por sistemas externos en tiempo real.

- StartEvent=CALL|Start|%{CallId}
  Default: CALL|Start|%{caller-ip}:%{caller-port}|%{callee-ip}:%{callee-port}|%{CallId}
  Define el evento que se va a visualizar para una nueva llamada. La cadena está parametrizada con los mismos parámetros de otros módulos de accounting. (Revisar la sección 9.4 ([SQLAacct]) para más detalle).

- StopEvent=CALL|Stop|%{CallId}
  Default: CALL|Stop|%{caller-ip}:%{caller-port}|%{callee-ip}:%{callee-port}|%{CallId}

Define el evento que se va a mostrar cuando una llamada finaliza (disconnected). La sitaxis y parámetros son los mismos que para StartEvent. Este evento es equivalente al antiguo evento CDR del puerto de estado, pero más fexible.

- UpdateEvent=CALL|Update|%{CallId}
  Default: CALL|Update|%{caller-ip}:%{caller-port}|%{callee-ip}:%{callee-port}|%{CallId}
  Define el evento utilizado para actualizar el estado actual de la llamada. La sintáxis y parámetros son los mismos que para StartEvent.
- ConnectEvent=CALL|Connect|%{CallId}
  Default: CALL|Connect|%{caller-ip}:%{caller-port}|%{callee-ip}:%{callee-port}|%{CallId}
  Define el evento cuando una llamada es establecida (connected). La sintaxis y parámetros son los mismos que para StartEvent.
- TimestampFormat=MySQL
  Default: N/A
  Formato de la cadena de fecha y hora utilizada en cada evento. Si esta configuración no se especifica, se utilizará la configuración global establecida en la sección principal del gatekeeper.

9.6. Sección [SyslogAcct]

Este módulo de accounting envía la información del accounting hacia el registro del sistema de Unix (Unix syslog) por consiguiente no está disponible para Windows.
El demonio local del "syslog.enrutará los mensajes de accounting de acuerdo a su configuración (generalmente /etc/syslog.conf).

- SyslogFacility=LOG_LOCAL1
  Default: LOG_USER
  Establezca la facilidad del syslog a uno de los siguientes valores: LOG_USER, LOG_DAEMON, LOG_AUTH, LOG_LOCAL0, LOG_LOCAL1, LOG_LOCAL2, LOG_LOCAL3, LOG_LOCAL4, LOG_LOCAL5, LOG_LOCAL6, LOG_LOCAL7.
- SyslogLevel=LOG_NOTICE
  Default: LOG_INFO
  Establezca el nivel de syslog a uno de los siguientes valores: LOG_EMERG, LOG_ALERT, LOG_CRIT, LOG_ERR, LOG_WARNING, LOG_NOTICE, LOG_INFO, LOG_DEBUG.
- StartEvent=CALL|Start|%{CallId}
  Default: CALL|Start|%{caller-ip}:%{caller-port}|%{callee-ip}:%{callee-port}|%{CallId}
  Define el evento mostrado para una nueva llamada. La cadena es parametrizada con las mismas variables utilizadas en otros módulos de accounting (Revisar 9.4 ([SQLAacct]) para más detalle).
- StopEvent=CALL|Stop|%{CallId}
  Default: CALL|Stop|%{caller-ip}:%{caller-port}|%{callee-ip}:%{callee-port}|%{CallId}
  Define el evento para cuando una llamada termina (disconnected). La sintaxis y parámetros son los mismos que para StartEvent. Este evento es equivalente al antiguo evento de puerto de estado CDR, pero más flexible.
- UpdateEvent=CALL|Update|%{CallId}
  Default: CALL|Update|%{caller-ip}:%{caller-port}|%{callee-ip}:%{callee-port}|%{CallId}
  Define el evento utilizado para actualizar el estado actual de la llamada. La sintaxis y parámetros son los mismos que para StartEvent.

- ConnectEvent=CALL|Connect|%{CallId}
  Default: CALL|Connect|%{caller-ip}:%{caller-port}|%{callee-ip}:%{callee-port}|%{CallId}
  Define el evento utilizado cuando una llamada es establecida (connected). La sintaxis y parámetros son los mismos que para StartEvent.

- TimestampFormat=MySQL
  Default: N/A
  Formato de la cadena de fecha y hora utilizada en cada evento. Si esta configuración no se especifica, se utilizará la configuración global establecida en la sección principal del gatekeeper.

9.6.1. Ejemplo de un esquema MySQL para el módulo SQLAcct

El módulo SQLAcct está diseñado para ser adaptado a cualquier estructura de base de datos que usted ya tenga. Aquí se puede definir todas las consultas para que encajen en las tablas que se hayan definido. A continuación se presenta un ejemplo de cómo implementar éste módulo empleando el motor de base de datos MySQL y puede ser utilizado como punto de partida para su propia implementación.

Creamos una nueva base de datos; aquí utilizaremos el nombre 'GNUGK':

```
create database GNUGK;
```

Luego creamos una tabla dentro de esta base de datos para almacenar sus datos de accounting; llamaremos a esta tabla con el nombre de 'CDR'.

```
create table GNUGK.CDR (
        gatekeeper_name varchar(255),
        call_number int zerofill,
        call_duration mediumint unsigned zerofill,
                index duration_idx (call_duration),
        disconnect_cause smallint unsigned zerofill,
                index dcc_idx (disconnect_cause),
        acct_session_id varchar(255),
        h323_id varchar(255),
        gkip varchar(15),
        CallId varchar(255),
        ConfID varchar(255),
        setup_time datetime,
        connect_time datetime,
        disconnect_time datetime,
        caller_ip varchar(15),
                index srcip_idx (caller_ip),
        caller_port smallint unsigned zerofill,
        callee_ip varchar(15),
                index destip_idx (callee_ip),
        callee_port smallint unsigned zerofill,
        src_info varchar(255),
        dest_info varchar(255),
        Calling_Station_Id varchar(255),
        Called_Station_Id varchar(255),
                index dialednumber_idx (Called_Station_Id (20)),
        Dialed_Number varchar(255)
);
```

Luego creamos un usuario que se encargará de manipular esta base de datos.

```
GRANT delete,insert,select,update ON GNUGK.* TO 'NombreQueUstedDesee'@'localhost' IDENTIFIED BY 'UnPasswor
```

Con este comando estaremos permitiendo el acceso solamente a los datos del servidor local. Si usted necesita acceder a estos datos desde cualquier otra computadora, se deben establecer las opciones de seguridad apropiadas.

Luego agregamos la siguiente configuración al archivo gnugk.ini para que inserte y actualice el historial de las llamadas dentro de la base de datos.

```
[Gatekeeper::Acct]
SQLAcct=optional;start,stop,update
FileAcct=sufficient;stop

[FileAcct]
DetailFile=Agregue aquí la ruta deseada. Algo como /var/log/cdr.log
StandardCDRFormat=0
CDRString=%g|%n|%d|%c|%s|%u|%{gkip}|%{CallId}|%{ConfId}|%{setup-time}|%{connect-time}|%{disconnect-time}|%
Rotate=daily
RotateTime=23:59

[SQLAcct]
Driver=MySQL
Database=GNUGK
Username=NombreQueUstedDesee
Password=UnPassword
StartQuery= insert into CDR (gatekeeper_name, call_number, call_duration, disconnect_cause, acct_session_i

StartQueryAlt= insert into CDR (gatekeeper_name, call_number, call_duration, disconnect_cause, acct_sessio

UpdateQuery= update CDR set call_duration=%d where gatekeeper_name='%g' and acct_session_id='%s'

StopQuery= update CDR set call_duration=%d, disconnect_cause=%c, disconnect_time='%{disconnect-time}' wher

StopQueryAlt= insert into CDR (gatekeeper_name, call_number, call_duration, disconnect_cause, acct_session

TimestampFormat=MySQL
```

10. Configuración de Vecinos

10.1. Sección [RasSrv::Neighbors]

Si el destino de un ARQ es desconocido, el gatekeeper envía LRQs hacia sus vecinos para preguntarles si ellos tienen este endpoint destino. Un vecino es seleccionado si uno de sus prefijos se corresponde con el destino o si éste tiene el prefijo "*". Se pueden especificar más de un prefijo. Usted puede utilizar caracteres especiales como "."' y "!"para emparejar mediante comodines (wildcards) y deshabilitar un prefijo específico.

Recíprocamente, el gatekeeper solamente contestará a los LRQs enviados desde los vecinos definidos en esta sección. Si usted especifica un prefijo vacío, ningún LRQ será enviado hacia ese vecino, pero el gatekeeper aceptará LRQs que vengan desde éste. Para especificar un prefijo vacío se añade un punto y coma a la entrada del vecino. Ejemplo:

```
GK1=192.168.0.5;
```

Si usted no pone el punto y coma, LRQs serán siempre enviados hacia este vecino.

El campo password es utilizado pata autenticar LRQs desde ese vecino. Revise la sección 8.1 ([Gatekeeper::Auth]) para más información.

Si una llamada que viene desde un vecino es aceptada o no depende también del parámetro AcceptNeighborsCalls configurado en la sección 5.1 ([RoutedMode]).

El manejo de los vecinos ha cambiado significativamente desde la versión 2.0 hasta la version 2.2. Actualmente los vecinos pueden ser configurados de dos maneras - de la forma anteriormente conocida y de una nueva manera.

Configuración en el formato anterior:

```
GKID=ip[:port;prefixes;password;dynamic]
```

Ejemplo:

```
GK1=192.168.0.5;*
GK2=10.0.1.1:1719;035,036;gk2
GK3=gk.citron.com.tw;;gk3;1
```

Configuración en el nuevo formato:

```
GKID="GnuGK"| ÇiscoGK"| ÇlarentGK"| "GlonetGK"
```

Ejemplo:

```
[RasSrv::Neighbors]
GK1=CiscoGK
GK2=GnuGK

[Neighbor::GK1]
GatekeeperIdentifier=GK1
Host=192.168.1.1
SendPrefixes=02
AcceptPrefixes=*
ForwardLRQ=always

[Neighbor::GK2]
GatekeeperIdentifier=GK2
Host=192.168.1.2
SendPrefixes=03,0048
AcceptPrefixes=0049,001
ForwardHopCount=2
ForwardLRQ=depends
```

El nuevo formato especifica dentro de la sección [RasSrv::Neighbors] solamente tipos de gatekeeper y las configuraciones para cada vecino se ubican en una sección separada.

10.2. Sección [RasSrv::LRQFeatures]

Define algunas características de los eventos LRQ y LCF.

- NeighborTimeout=1
  Default: 2
  Tiempo de espera en segundos para esperar respuestas desde los vecinos. Si no hay respuesta desde todos los vecinos después de este tiempo, el gatekeeper contestará con un ARJ al endpoint que envió el ARQ.

- ForwardHopCount=2
  Default: N/A
  Si el gatekeeper recive un LRQ en el que el destino es desconocido, éste reenviará este mensaje hacia sus vecinos. Cuando el gatekeeper recibe un LRQ y decide que el mensaje debe ser reenviado hacia otro gatekeeeper, éste, primero decrementa el campo hopCount del mensaje LRQ. Si hopCount ha llegado a 0, el gatekeeper no reenviará el mensaje. Esta opción define el número de gatekeepers a través de los cuales un mensaje LRQ puede propagarse. Tenga en cuenta que esto solo afecta al remitente del LRQ, no al receptor. Este ajuste puede ser superpuesto con la configuración de un vecino particular.

- AlwaysForwardLRQ=1
  Default: 0
  Forzar a que el gatekeeper reenvie un LRQ incluso si no hay hopCount en el LRQ. Para evitar LRQ repetidos, usted debe utilizar esta opción muy cuidadosamente. Esta opción es utilizada solamente para estilos anteriores de configuración de vecinos (Version 2.0), el nuevo estilo lee las configuraciones desde la sección de configuración de un vecino específico.

- AcceptForwardedLRQ=1
  Default: 1
  Definir si aceptar o no un LRQ reenviado desde los vecinos. Esta configuración puede ser superpuesta o deshabilitada con la configuración de un vecino particular.

- ForwardResponse=0
  Default: 0
  Si el gatekeeper reenvia los mensajes LRQ recibidos, éste puede decidir entre recibir la repuesta LCF o permitir que ésta regrese directamente hacia el gatekeeper que originó en LRQ. Establezca esta opción en 1, si el gatekeeper necesita recibir respuestas LCF de los LRQs reenviados. Este ajuste puede ser superpuesto o deshabilitado con la configuración de un vecino particular.

- ForwardLRQ=always | never | depends
  Default: depends
  Esta configuración determina si el LRQ recibido debería o no ser reenviado. Si se establece en always, reenvía LRQs de manera incondicional, si es never, bloquea los LRQ reenviados, y si es depends le indica al gatekeeper que reenviará LRQ solamente si su contador de saltos (hop count) es mayor a 1. Esta configuración puede ser superpuesta o deshabilitada con la configuración de un vecino particular.

- AcceptNonNeighborLRQ=1
  Default: 0
  Definir si aceptar o no un mensaje LRQ enviado desde GKs que no han sido definidos

como vecinos. Eso puede ser utilizado con la política de enrutado "SRV routing policy"para ubicar llamadas hacia GKs de terceros. Esto debe ser utilizado conjuntamente con una política "LRQ Authentication".

10.2.1. Sección [Neighbor::...]

Las secciones que empiezan con [Neighbor:: son para las configuraciones especificas de un vecino.

- GatekeeperIdentifier=GKID
  Default: N/A
  Identificador del Gatekeeper para este vecino. Si esta opción no se especifica, el identificador es tomado de la segunda parte del nombre de esta sección Neighbor::.
- Host=192.168.1.1
  Default: N/A
  Una dirección IP para este vecino.
- Password=secret
  Default: N/A
  Un password que se utilizará pata validar los crypto tokens recibidos desde los LRQs entrantes. Esto aún no está implementado.
- Dynamic=0
  Default: 0 1 significa que la dirección IP para este vecino puede cambiar.
- SendPrefixes=004,002:=1,001:=2
  Default: N/A
  Una lista de prefijos que este vecino espera recibir para los LRQs. Si se especifica '*', Los LRQs siempre serán enviados hacia este vecino. Se puede asignar una prioridad a cada prefijo para cada vecino (usando la sintaxis :=), de esta manera en caso de que hayan demasiados LCF recibidos de demasiados vecinos, aquel con la prioridad más alta será elegido para rutear la llamada. Uno también puede dirigir el gatekeeper para que envíe LRQs hacia este vecino basándose en el tipo de alias:
  SendPrefixes=h323_ID,dialedDigits,001
- AcceptPrefixes=*
  Default: *
  Una lista de prefijos que el gatekeeper aceptará en los LRQs recibidos desde este vecino. Si se especifica '*' , todos los LRQs serán aceptados desde este vecino. Uno puede también dirigir al gatekeeper para que acepte LRQ desde este vecino basándose en el tipo de alias:
  AcceptPrefixes=dialedDigits
- ForwardHopCount=2
  Default: N/A
  Si el gatekeeper recibe un LRQ en el que el destino es desconocido, éste podría reenviar este mensaje hacia sus vecinos. Cuando el gatekeeper recibe un LRQ y decide que el mensaje debe ser reenviado hacia otro gatekeeper, éste primero decrementa el campo hopCount del LRQ. Si hopCount ha llegado a 0, el gatekeeper no reenviará el mensaje. Esta opción define el número de gatekeepers a través de los cuales un LRQ

puede propagarse. Tenga en cuenta que ésto solo afecta al remitente del LRQ, no al receptor.

- AcceptForwardedLRQ=1
  Default: 1
  Definir si aceptar o no un LRQ reenviado desde este vecino.

- ForwardResponse=0
  Default: 0
  Si el gatekeeper reenvió los mensajes LRQ recibidos, éste puede decidir entre recibir la repuesta LCF o permitir que ésta regrese directamente hacia el gatekeeper que originó en LRQ. Establezca esta opción en 1, si el gatekeeper necesita recibir respuestas LCF de los LRQs reenviados.

- ForwardLRQ=always | never | depends
  Default: depends
  Esta configuración determina si el LRQ recibido deberá o no ser reenviado. Si es always, reenvía LRQs de manera incondicional, si es never, bloquea los LRQ reenviados ys is es depends le indica al gatekeeper que reenviará LRQ solamente si su contador de saltos (hop count) es mayor a 1. Este ajuste puede ser superpuesto o deshabilitado con la configuración de un vecino particular.

11. Configuración Por-Endpoint

Adicionalmente a las opciones estándar del fichero de configuración, se puede especificar atributos de configuración por-endpoint en el fichero de configuración. La sintaxis es como sigue:

11.1. Sección [EP::...]

```
[EP::ALIAS]
Key Name=Value String
```

ALIAS es reemplazado con un alias actual para el endpoint al cual se va a aplicar la configuración. Actualmente, están reconocidas las siguientes opciones:

- Capacity=10
  Default: -1
  Capacidad de llamadas para un endpoint. No más de Capacity llamadas concurrentes serán enviadas a este endpoint. En caso de puertas de enlace (gateways), si más de un gateway se corresponde con un número marcado, la llamada será enviada al primer gateway disponible (que tenga suficiente capacidad).

- GatewayPriority=1
  Default: 1
  Aplicado solo para gateways. Permite la prioridad basada en casos de encaminado, cuando más de un gateway se corresponde con un número marcado. El valor más pequeño es la prioridad más alta asignada a un gateway. Una llamada es encaminada hacia el primer gateway disponible (que tenga capacidad disponible) con la prioridad más alta (los valores más bajos en GatewayPriority).

- GatewayPrefixes=0048,0049,0044
  Default: N/A
  Prefijos adicionales para este gateway. Aplicado solo a gateways. Los caracteres especiales . y ! pueden ser usados aquí para la correspondencia con cualquier dígito y desactivar el prefijo (respectivamente).
- CalledTypeOfNumber=1
  Default: N/A
  Fija el tipo de número para Called-Party-Number a un valor específicado por las llamadas enviadas a este endpoint (0 - UnknownType (Tipo Desconocido), 1 - InternationalType (Tipo Internacional), 2 - NationalType (Tipo Nacional), 3 - NetworkSpecificType (Tipo Red Específico), 4 - SubscriberType (Tipo Suscriptor), 6 - AbbreviatedType (Tipo Abreviado), 7 - ReservedType (Tipo Reservado)).
- CallingTypeOfNumber=1
  Default: N/A
  Fija el tipo de número para Calling-Party-Number a un valor especificado por las llamadas enviadas a este endpoint (0 - UnknownType (Tipo Desconocido), 1 - InternationalType (Tipo Internacional), 2 - NationalType (Tipo Nacional), 3 - NetworkSpecificType (Tipo Red Específico), 4 - SubscriberType (Tipo Suscriptor), 6 - AbbreviatedType (Tipo Abreviado), 7 - ReservedType (Tipo Reservado)).
- Proxy=1
  Default: 0
  Habilita o deshabilita el "proxiado"de llamadas enviadas a este endpoint (0 - no cambia las configuraciones globales del proxy, 1 - forza el modo proxy, 2 - deshabilita el modo proxy).

Ejemplo:

```
[RasSrv::PermanentEndpoints]
192.168.1.1=gw1;48
192.168.1.2=gw2;48,!4850,!4860,!4869,!4888

[EP::gw1]
Capacity=60
GatewayPriority=1

[EP::gw2]
Capacity=30
GatewayPriority=2
```

En este ejemplo, la llamada será enviada al gateway gw1 a menos que su capacidad esté completamente utilizada (60 llamadas concurrentes) y luego al gateway gw2.

12. Configuración Avanzada

12.1. Sección [CallTable]

- GenerateNBCDR=0
  Default: 1
  Genere CDRs para llamadas que vienen desde zonas vecinas. La dirección IP y el

ID del endpoint que está llamando se presenta como una cadena vacía. Esto es frecuentemente utilizado para propósitos de depuración.

- GenerateUCCDR=0
  Default: 0
  Genere CDRs para llamadas que son desconectadas (unconnected). Esto es frecuentemente utilizado para propósitos de depuración. Tenga presente que una llamada se considera desconectada (unconnected) solamente si el gatekeeper utiliza modo enrutado (routed mode) y no recibe un mensaje de conexión Q.931 (Q.931 Connect). En modo directo (direct mode), siempre una llamada es considerada conectada (connected).

- DefaultCallDurationLimit=3600
  Default: 0
  Límite máximo de tiempo (por defecto) que durará una llamada (en segundos). Establesca esta variable a 0 para deshabilitar esta característica y no limitar la duración de las llamadas.

- AcctUpdateInterval=60
  Default: 0
  Un intervalo de tiempo (en segundos) para que la actualización de accounting (accounting update) se registre para cada llamada en progreso. Detalles exactos de actualización de accounting (accounting update) dependen del módulo de registro de accounting elegido (ver la sección 9.1 ([Gatekeeper::Acct])). En general, la actualización del accounting (accounting update) es con la finalidad de tener un servicio de respaldo con el incremento de la duración de la llamada, para llamadas establecidas (connected calls). El valor por defecto 0 le indica al gatkeeper que no realize actualización de accounting en absoluto. Por favor tenga presente que establecer períodos cortos de tiempo podría disminuír el rendimiento del GK.

- TimestampFormat=Cisco
  Default: RFC822
  Formato de presentación de las cadenas de fecha y hora impresas dentro del los CDRs.

- IRRFrequency=60
  Default: 120
  Establezca la "irrFrequency.en los mensajes ACF. 0 deshabilita esta opción.

- IRRCheck=TRUE
  Default: FALSE
  Revisa si ambos endpoints en una llamada envían los IRRs pedidos. Una llamada terminará si uno de los endpoints no envía un IRR después de 2 * irrFrequency.

- SingleFailoverCDR=FALSE
  Default: TRUE
  Cuando failover está activo, se puede probar más de un gateway para establecer una llamada. Este cambio define si uno o varios CDRs se generan para una llamada.

12.2. Sección [Endpoint]

El gatekeeper puede trabajar como un endpoint registrandose con otro gatekeeper. Con esta característica, usted puede construír facilmente jerarquías de gatekeepers. Esta sección define las características de un endpoint para el gatekeeper.

- Gatekeeper=10.0.1.1
  Default: no
  Define un gatekeeper padre (parent gatekeeper) para el endpoint(el gatekeeper), con el cual se va a registrar. No trate de registrarse con usted mismo, a menos de que usted desee confundirse. Para dehabilitar esta característica, configure este campo para que sea no.

- Type=Gateway
  Default: Gateway
  Defina el tipo de terminal para el endpoint. Los valores válidos son Gateway o Terminal.

- Vendor=Cisco | GnuGk | Generic
  Default: GnuGk
  Seleccione el tipo de gatekeeper padre para habilitar la extensiones específicas de un vendedor.

- H323ID=CitronProxy
  Default: <Name>
  Especifique el o los alias H.323 ID para el endpoint. Múltiples alias pueden separarse con comas.

- E164=18888600000,18888700000
  Default: N/A
  Define el alias E.164 (dialedDigits) para el endpoint. Múltiples alias pueden separarse con comas.

- Password=123456
  Default: N/A
  Especifique un password para que sea enviado al gatekeeper padre. Todas las peticiones RAS contendrán el password en el campo cryptoTokens (MD5 & HMAC-SHA1-96) y en el campo tokens (CAT). Para enviar peticiones RAS sin los camposcryptoTokens y tokens, establesca el password como vacío. Si la opción EncryptAllPasswords está habilitada, o la variable KeyFilled está definida en esta sección, el password está en forma encriptada y deberá ser creada utilizando la herramienta addpasswd.

  Además, el password es también utilizado en mensajes LRQs enviados a gatekeepers vecinos.

- Prefix=188886,188887
  Default: N/A
  Registre los prefijos especificados con el gatekeeper padre. Solamente tiene efecto cuando el tipo de endpoint es Gateway.

- TimeToLive=900
  Default: 60
  Sugiera un valor time-to-live (en segundos)para el proceso de registro. Tenga presente que el contador real para time-to-live es asignado por el gatekeeper padre en el mensaje RCF contestando a la petición RRQ.

- RRQRetryInterval=10
  Default: 3
  Defina un intervalo de reintento en segundos para reenviar un RRQ si no se ha recibido respuesta desde el gatekeeper padre. Este intervalo es duplicado con cada fallo, a un máximo de RRQRetryInterval * 128.

- ARQTimeout=2
  Default: 2
  Define un valor de espera en segundos para los ARQs.
- UnregisterOnReload=1
  Default: 0
  Define si el gatekeeper hijo se desregistra y re-registra con su gatekeeper padre cuando recibe el comando Reload.
- NATRetryInterval=60
  Default: 60
  Tiempo que se debe esperar antes de tratar de reconectar el TCP NAT signalling socket (en segundos). Esto puede pasar por cualquiera de estas razones: la conexión no puede ser establecida o ésta ha sido rota.
- NATKeepaliveInterval=86400
  Default: 86400
  Define que tan a menudo es refrescada la conexión de señalización TCP NAT con un gatekeeper padre. Puesto que las NAT box usualmente mantienen TCP mappings solamente por un tiempo definido, es bueno establecer éste, con un valor un poco mas pequeño que el de la NAT box. El refresco de la conexión es realizado enviando un mensaje Q.931 IncomingCallProceeding. Si su NAT realiza traducción de puertos TCP, usted podría necesitar establecer éste a un valor cercano a 60 segundos.
- Discovery=0
  Default: 1
  Decida si descubre o no el gatekeeper padre enviando primero un GRQ.
- UseAlternateGK=0
  Default: 1
  Habilite la característica de alternar gatekeepers. Si los mensajes GRJ/GCF/RFC recibidos desde el gatekeeper padre contienen una lista de gatekeepers alternos, esta información es almacenada y puede ser utilizada después para registrarse con otro gatekeeper en caso de cualquier falla. Si usted no desea usar esta característica, establezca esta variable a 0.
- GatekeeperIdentifier=ParentGK
  Default: Not set
  Defina este parámetro si usted desea aceptar solamente aquellos gatekeepers que correspondan a este identificador de gatekeeper. Es muy utilizado con GRQ discovery y puede prevenir una asignación de gatekeeper accidental. No establezca esta variable, si a usted no le importa acerca de identificadores de gatekeeper o si usted utiliza gatekeepers alternos que pueden tener diferentes identificadores de gatekeeper establecidos.
- EndpointIdentifier=OpenH323GK
  Default: Not set
  Establezca esta característica si usted desea utilizar un identificador de endpoint específico para este gatekeeper hijo. Si esta opción no esta establecida (por defecto), el gatekeeper padre asignará el identificador en un mensaje GCF/RCF.

12.3. Sección [CTI::Agents]

Esta sección permite la configuración de las llamadas colas virtuales (virtual queues) para permitir distribución de llamadas entrantes por una aplicación externa, mediante el puerto de estado. Una cola virtual tiene un alias H.323 que puede ser llamado como un endpoint.

En el arrivo de un mensaje ARQ a una cola virtual, el gatekeeper señala una RouteRequest en el puerto de estado y espera a que una aplicación externa responda ya sea con un RouteReject (entonces el ARQ será rechazado) o con un RouteToAlias/RouteToGateway el cual conduce al ARQ a ser reescrito de este modo la llamada será ruteada hacia el alias (eg. call center agent) especificado por la aplicación externa.

Si no se recibe ninguna respuesta después de un período de tiempo, la llamada es finalizada.

Usted puede definir colas virtuales de tres maneras:

- nombre exacto de alias - Una lista de alias es asignada. Si un alias destino ARQ corresponde a uno de estos nombres, la cola virtual es activada,
- prefijo - Una lista de prefijos es dada. Si un alias destino ARQ inicia con uno de estos prefijos, la cola virtual es activada,
- expresión regular - Una expresión regular es dada. Si un alias destino ARQ corresponde a la expresión, la cola virtual es activada.

Ver la sección de monitoreo (monitoring section) para detalles sobre los mensajes y respuestas.

- VirtualQueueAliases
  Default: none
  Este parámetro define una lista de alias H.323 para las colas virtuales (Utilizado con la vqueue RoutingPolicy).

  Ejemplo:

  ```
  VirtualQueueAliases=sales,support
  ```

- VirtualQueuePrefixes
  Default: none
  Este parámetro define una lista de prefijos para las colas virtuales (Utilizado con la vqueue RoutingPolicy).

  Ejemplo:

  ```
  VirtualQueuePrefixes=001215,1215
  ```

- VirtualQueueRegex
  Default: none
  Este parámetro define una expresión regular para las colas virtuales (Utilizado con la vqueue RoutingPolicy).

  Ejemplo (números que empiezan con 001215 o 1215):

  ```
  VirtualQueueRegex=^(001|1)215[0-9]*$
  ```

- RequestTimeout
  Default: 10
  Tiempo de espera en segundos para que la aplicación externa responda el RouteRequest. Si no se recibe ninguna respuesta durante este tiempo un ARJ será enviado hacia el enpoint que llama (caller).

12.4. Sección [SQLConfig]

Carge configuraciones del gatekeeper desde una base de datos SQL (adicionalmente a las configuraciones leidas desde el archivo de configuración). Una ConfigQuery genérica puede ser utilizada para leer casi todas las configuraciones desde la base de datos y/o una de [RasSrv::RewriteE164], [RasSrv::PermanentEndpoints], [RasSrv::Neighbors], [RasSrv::GWPrefixes] consultas pueden ser utilizadas para cargar configuraciones particulares. Los valores leídos desde la base de datos SQL tiene preferencia sobre las configuraciones encontradas en el archivo de configuración.

- Driver=MySQL | PostgreSQL | Firebird
  Default: N/A
  El driver SQL a utilizar. Actualmente, drivers MySQL, PostgreSQL y Firebird están implementados.
- Host=DNS[:PORT] | IP[:PORT]
  Default: localhost
  Dirección del host servidor SQL. Puede estar en la forma de DNS[:PORT] o IP[:PORT]. Como sql.mycompany.com o sql.mycompany.com:3306 o 192.168.3.100.
- Database=billing
  Default: billing
  El nombre de la base de datos a la cual se realiza la conexión.
- Username=gnugk
  El nombre de usuario utilizado para conectarse a la base de datos.
- Password=secret
  El password utilizado para conectarse a la base de datos. Si el password no está especificado, un intento de conexión a la base de datos sin un password es realizada. Si EncryptAllPasswords está habilitada, o una variable KeyFilled esta definida en esta sección, el password esta en forma encriptada y puede ser creado utilizando la herramientas addpasswd.
- ConfigQuery=SELECT ...
  Default: N/A
  Defina una consulta SQL utilizada para leer configuraciones del gatekeeper desde la base de datos. La consulta es parametrizada, lo que significa que el remplazo de parámetros ocurre antes de que la consulta es ejecutada. Los parámetros están denotados por las cadenas %1, %2, ... Especifique %% para insertar un caracter de porcentaje antes de un dígito dentro de una cadena (así %%1), especifique %{1} para pemitir expansión dentro de expesiones complejas como %{1}123. Para ConfigQuery solamente está definido un parámetro:
  - %1 - El identificador del gatekeeper

  Se espera que la consulta devuelva cero o mas filas de datos, en donde cada fila consiste de tres columnas:

- columna con índice 0 - nombre de la sección de configuración (config section name)
- columna con índice 1 - nombre de la opción (config key)
- columna con índice 2 - valor de la opción (config value)

Ejemplos de consultas:

```
ConfigQuery=SELECT secname, seckey, secval FROM sqlconfig WHERE gk = '%1'
ConfigQuery=SELECT 'RasSrv::RRQAuth', alias, rule FROM rrqauth WHERE gk = '%1'
```

- RewriteE164Query=SELECT ...
  Default: N/A
  Defina una consulta SQL para recuperar desde la base de datos reglas de reescritura (rewrite rules) para la sección [RasSrv::RewriteE164]. La consulta es parametrizada, lo que significa que el reemplazo de parámetros ocurre antes de que cada consulta es ejecutada. Los parámetros están denotados por cadenas %1, %2, ... Especifique %% para insertar un caracter de porcentaje antes de un dígito dentro de una cadena (así %%1), especifique %{1} para pemitir expansión dentro de expesiones complejas como %{1}123. Para RewriteE164Query solamente está definido un parámetro:

  - %1 - El identificador del gatekeeper

  Se espera que la consulta devuelva cero o mas filas de datos, en donde cada fila consiste de dos columnas:

  - columna con índice 0 - nombre de la regla de reescritura (rewrite rule key)
  - columna con índice 1 - valor de la regla de reescritura (rewrite rule value)

  Ejemplos de consulta:

  ```
  RewriteE164Query=SELECT rkey, rvalue FROM rewriterule WHERE gk = '%1'
  ```

- NeighborsQuery=SELECT ...
  Default: N/A
  Defina una consulta SQL para recuperar desde la base de datos entradas vecinas (neighbor entries) para la sección [RasSrv::Neighbors]. La consulta es parametrizada, lo que significa que el reemplazo de parámetros ocurre antes de que cada consulta es ejecutada. Los parámetros están denotados por cadenas %1, %2, ... Especifique %% para insertar un caracter de porcentaje antes de un dígito dentro de una cadena (así %%1), especifique %{1} para pemitir expansión dentro de expesiones complejas como %{1}123. Para NeighborsQuery está definido un parámetro:

  - %1 - El identificador del gatekeeper

  Se espera que la consulta devuelva cero o mas filas de datos, en donde cada fila consiste de seis columnas:

  - columna con índice 0 - nombre del vecino (identificador)
  - columna con índice 1 - dirección IP del vecino
  - columna con índice 2 - número de puerto del vecino
  - columna con índice 3 - prefijos opcionales (separados por comas)
  - columna con índice 4 - password opcional
  - columna con índice 5 - IP dinámica opcional (optional dynamic IP flag)

  Ejemplos de consultas:

```
NeighborsQuery=SELECT nid, nip, nport, npfx, NULL, 0 FROM neighbor WHERE gk = '%1'
```

- PermanentEndpointsQuery=SELECT ...
  Default: N/A
  Defina una consulta SQL para recuperar endpoints permanentes desde la base de datos para la sección [RasSrv::PermanentEndpoints]. La consulta es parametrizada, lo que significa que el reemplazo de parámetros ocurre antes de que cada consulta es ejecutada. Los parámetros están denotados por cadenas %1, %2, ... Especifique %% para insertar un caracter de porcentaje antes de un dígito dentro de una cadena (así %%1), especifique %{1} para pemitir expansión dentro de expesiones complejas como %{1}123. Para PermanentEndpointsQuery solamente está definido un parámetro:
  - %1 - El identificador del gatekeeper

  Se espera que la consulta devuelva cero o mas filas de datos, en donde cada fila consiste de cuatro columnas:
  - columna con índice 0 - dirección IP del endpoint permanente
  - columna con índice 1 - número de puerto del endpoint permanente
  - columna con índice 2 - alias del endpoint permanente
  - columna con índice 3 - prefijos opcionales del endpoint permanente (separados por comas)

  Ejemplos de consultas:

  ```
  PermanentEndpointsQuery=SELECT peip, 1720, pealias, NULL FROM permanentep WHERE gk = '%1'
  ```

- GWPrefixesQuery=SELECT ...
  Default: N/A
  Defina una consulta SQL para recuperar prefijos de gateways desde la base de datos para la sección [RasSrv::GWPrefixes]. La consulta es parametrizada, lo que significa que el reemplazo de parámetros ocurre antes de que cada consulta es ejecutada. Los parámetros están denotados por cadenas %1, %2, ... Especifique %% para insertar un caracter de porcentaje antes de un dígito dentro de una cadena (así %%1), especifique %{1} para pemitir expansión dentro de expesiones complejas como %{1}123. Para GWPrefixesQuery solamente está definido un parámetro:
  - %1 - El identificador del gatekeeper

  Se espera que la consulta devuelva cero o mas filas de datos, en donde cada fila consiste de dos columnas:
  - columna con índice 0 - alias del gateway
  - columna con índice 1 - prefijos del gateway (separados por comas)

  Ejemplos de consultas:

  ```
  GWPrefixesQuery=SELECT gwalias, gwpfx FROM gwprefix WHERE gk = '%1'
  ```

13. Monitoreando el Gatekeeper

13.1. Puerto de Estado

El puerto de estado es una interfaz externa para monitorear y controlar el gatekeeper. El gatekeeper arrojará mensajes sobre las llamadas actuales hacia todos los clientes conectados y puede recibir comandos mediante esta interfaz.

Los mensajes enviados por el gatekeeper hacia el puerto de estado estan agrupados dentro de tres Niveles de rastreo de salida:

- Nivel 0

  Notificaciones de recarga (Reload) y respuesta directa a comandos ejecutados.

- Nivel 1

  Notificaciones de recarga (Reload), respuesta directa a comandos ejecutados, CDRs y Route Requests.

- Nivel 2

  Todas las salidas (Notificacione se recarga (Reload), respuesta directa a comandos ejecutados, CDRs, Route Requests,RAS, ...). Éste es el nivel de salida por defecto.

El cliente que esté conectado hacia el puerto de estado puede seleccionar el nivel de salida (output level) que él desee.

La interfaz es un simple puerto TCP (por defecto: 7000), usted puede conectarse hacia éste mediante telnet u otro cliente. Un ejemplo de un cliente es el Java GUI, aka GkGUI. Otro ejemplo es la aplicación Automatic Call Distribution, aka GnuGk ACD.

13.1.1. Áreas de aplicación.

Depende de usted lo que haga con el poder del puerto de estado, pero aqui hay algunas ideas:

- Monitoreo de llamadas
- Monitoreo de endpoints registrados
- Interfaz Gráfica de usuario (GUI)

  Ver GkGUI.

- Ruteo de llamadas

  Ver GnuGk ACD.

- Aplicaciones de facturación

  Analice los mensajes CDR y adminístrelos luego desde una aplicación de facturación.

- Interfaz con aplicaciones externas

  Si usted no desea publicar el código fuente de alguna característica adicional, simplemente publique la funcionalidad central y únalo mediante el puerto de estado y mantenga la parte externa como privada.

13.1.2. Ejemplos:

Supongamos que usted esta interesado en los CDRs (call details records) y quiere un proceso que trabaje en modo batch en intervalos regulares.

Aqui está un script simple escrito en Perl (gnugk_cdr.pl) que inicia el gatekeeper y además conecta un simple cliente para la Status Interface y registra los CDRs dentro de un logfile. Usted debería modificar un poco este script para ajustarlo a sus necesidades.

```
#!/usr/bin/perl
# sample program that demonstrates how to write the CDRs to a log file
use strict;
use IO::Socket;
use IO::Handle;

my $logfile = "/home/jan/cdr.log";      # CHANGE THIS
my $gk_host = "localhost";
my $gk_port = 7000;
my $gk_pid;

if ($gk_pid = fork()) {
        # parent will listen to gatekeeper status
        sleep(1);       # wait for gk to start
        my $sock = IO::Socket::INET->new(PeerAddr => $gk_host, PeerPort => $gk_port, Proto => 'tcp')
        if (!defined $sock) {
                die "Can't connect to gatekeeper at $gk_host:$gk_port";
        }
        $SIG{HUP} = sub { kill 1, $gk_pid; };   # pass HUP to gatekeeper
        $SIG{INT} = sub { close (CDRFILE); kill 2, $gk_pid; };  # close file when terminated

        open (CDRFILE, ">>$logfile");
        CDRFILE->autoflush(1);  # don't buffer output
        while (!$sock->eof()) {
                my $msg = $sock->getline();
                $msg = (split(/;/, $msg))[0];   # remove junk at end of line
                my $msgtype = (split(/\|/, $msg))[0];
                if ($msgtype eq "CDR") {
                        print CDRFILE "$msg\n";
                }
        }
        close (CDRFILE);
} else {
        # child starts gatekeeper
        exec("gnugk");
}
```

Tenga siempre presente que éste es justamente un ejemplo para mostrar el uso del puerto de estado. Usted puede utilizar el módulo FileAcct para registrar los CDRs en un sistema en producción.

13.1.3. GUI para el gatekeeper

Hay algunas Graphical User Interface (GUI) frontends para el Gatekeeper.

- Java GUIDesarrollado por Jan Willamowius. Usted puede monitorear los registros y llamadas que están en el gatekeeper. Un clic derecho sobre un botón le muestra a usted un menú desplegable con información de dicho endpoint.

  Esta GUI trabaja con Java 1.0 soportado por muchos navegadores web. Por razones de seguridad en GUI debe ser ejecutado como una aplicación standalone o administrada por un servidor web sobre el mismo número de IP del gatekeeper (usted no puede ejecutar esta GUI como un applet mediante el archivo local).

  La aplicación está disponible en *GnuGk Java GUI* <http://www.gnugk.org/h323gui.html>

- GkGUIUna nueva aplicación standalone Java desarrollado por *Citron Network Inc.* <http://www.citron.com.tw/> Este requiere Java 1.4. Entre las nuevas características se incluyen:

  - Monitorea múltiples gatekeepers simultáneamente.
  - Dos modos de vista: Button List y Tree List.
  - Call Detail Record(CDR) y estadísticas.
  - GK Status Log.
  - Colores diferentes para diferentes tipos de endpoints.
  - Modifica la configuración del gatekeeper.
  - Forza a desregistrar endpoints.
  - Guarda e imprime el estado del log y del CDR.

  El GkGUI está lanzada bajo GNU General Public License, disponible en *GnuGk Development* <http://www.gnugk.org/h323develop.html#java>

13.2. Comandos de Referencia

Esta sección muestra todos los comandos que usted puede enviar al puerto de estado (manualmente o mediante una aplicación externa). Todos los comandos son case-insensitive. Pero algunos parámetros pueden ser sensibles a mayúsculas.

El comando help o h le mostrará una lista de todos los comando disponibles.

- Reload Recarga o reinicia la configuración.

- Version, v Muestra la versión e información del Sistema Operativo donde está ejecutándose el gatekeeper.

- Statistics, s Muestra información estadística del gatekeeper.

  Ejemplo:

```
Statistics
-- Endpoint Statistics --
Total Endpoints: 21  Terminals: 17  Gateways: 4  NATed: 2
Cached Endpoints: 1  Terminals: 1  Gateways: 0
-- Call Statistics --
Current Calls: 1 Active: 1 From Neighbor: 0 From Parent: 0
```

```
Total Calls: 1539  Successful: 1076  From Neighbor: 60  From Parent: 5
Startup: Fri, 21 Jun 2002 10:50:22 +0800   Running: 11 days 04:22:59
;
```

- PrintAllRegistrations, r, ? Muestra todos los endpoints registrados.

  Formato:

```
AllRegistrations
RCF|IP:Port|Aliases|Terminal_Type|EndpointID
...
Number of Endpoints: n
;
```

  Ejemplo:

```
AllRegistrations
RCF|10.1.1.10:1720|800:dialedDigits=Wei:h323_ID|terminal|1289_endp
RCF|10.0.1.43:1720|613:dialedDigits=Jacky Tsai:h323_ID|terminal|1328_endp
RCF|10.0.1.55:1720|705:dialedDigits=Sherry Liu:h323_ID|terminal|1333_endp
Number of Endpoints: 3
;
```

- PrintAllRegistrationsVerbose, rv, ?? Muestra el detalle de todos los endpoints registrados.

  Formato:

```
AllRegistrations
RCF|IP:Port|Aliases|Terminal_Type|EndpointID
Registration_Time C(Active_Call/Connected_Call/Total_Call) <r>
[Prefixes: ##] (gateway only)
...
Number of Endpoints: n
;
```

  Ejemplo:

```
AllRegistrations
RCF|10.0.1.8:1720|Accel-GW2:h323_ID|gateway|1322_endp
Wed, 26 Jun 2002 16:40:03 +0800 C(1/5/33) <1>
Prefixes: 09,002
RCF|10.1.1.10:1720|800:dialedDigits=Wei:h323_ID|terminal|1289_endp
Wed, 26 Jun 2002 16:40:55 +0800 C(0/32/39) <1>
RCF|10.0.1.66:1720|716:dialedDigits=Vicky:h323_ID|terminal|1425_endp
Wed, 26 Jun 2002 16:40:58 +0800 C(1/47/53) <1>
Number of Endpoints: 2
;
```

- PrintCurrentCalls, c, ! Muestra todas las llamadas actuales utilizando la misma sintaxis de ACF como en el establecimiento de llamada.

  Formato:

```
CurrentCalls
Call No. # | CallID | Call_Duration | Left_Time
Dialed_Number
ACF|Caller_IP:Port|Caller_EPID|CRV|DestinationInfo|SrcInfo|IsAnswered;
ACF|Callee_IP:Port|Callee_EPID|CRV|DestinationInfo|SrcInfo|IsAnswered;
...
Number of Calls: Current_Call Active: Active_Call From Neighbor: Call_From_Neighbor \
```

```
From Parent: Call_From_Parent
;
```

Ejemplo:

```
CurrentCalls
Call No. 29 | CallID bd c6 17 ff aa ea 18 10 85 95 44 45 53 54 77 77 | 109 | 491
Dial 0953378875:dialedDigits
ACF|10.0.1.49:1720|4048_CGK1|25263|frank:h323_ID|gunter:h323_ID|false;
ACF|10.1.1.1:1720|4037_CGK1|25263|gunter:h323_ID|frank:h323_ID|true;
Call No. 30 | CallID 70 0e dd c0 9a cf 11 5e 00 01 00 05 5d f9 28 4d | 37 | 563
Dial 0938736860:dialedDigits
ACF|10.0.1.48:1032|4041_CGK1|11896|sue:h323_ID|peter:h323_ID|false;
ACF|10.1.1.1:1720|4037_CGK1|11896|peter:h323_ID|sue:h323_ID|true;
Number of Calls: 2 Active: 2 From Neighbor: 0 From Parent: 0
;
```

- PrintCurrentCallsVerbose, cv, !! Muestra el detalle de todas las llamadas actuales.

  Formato:

```
CurrentCalls
Call No. # | CallID | Call_Duration | Left_Time
Dialed_Number
ACF|Caller_IP:Port|Caller_EPID|CRV|DestinationInfo|SrcInfo|IsAnswered;
ACF|Callee_IP:Port|Callee_EPID|CRV|DestinationInfo|SrcInfo|IsAnswered;
# Caller_Aliases|Callee_Aliases|Bandwidth|Connected_Time <r>
...
Number of Calls: Current_Call Active: Active_Call From NB: Call_From_Neighbor
;
```

  Ejemplo:

```
CurrentCalls
Call No. 48 | CallID 7d 5a f1 0a ad ea 18 10 89 16 00 50 fc 3f 0c f5 | 30 | 570
Dial 0225067272:dialedDigits
ACF|10.0.1.200:1720|1448_endp|19618|frank:h323_ID|gunter:h323_ID|false;
ACF|10.0.1.7:1720|1325_endp|19618|gunter:h323_ID|frank:h323_ID|true;
# Sherry:h323_ID|Accel-GW1:h323_ID|200000|Wed, 26 Jun 2002 17:29:55 +0800 <2>
Number of Calls: 1 Active: 1 From NB: 0
;
```

- Find, fnewline> Busca un endpoint registrado mediante el alias o un prefijo. Para buscar una alias de un tipo específico (h323_ID, dialedDigits), anteponga el tipo de alias (h323, e164, url, email) al alias, seguido por dos puntos (:).

  Formato:

```
Find Alias
RCF|IP:Port|Aliases|Terminal_Type|EndpointID
;
```

  Ejemplo:

```
f 800
RCF|10.1.1.10:1720|800:dialedDigits=Wei:h323_ID|terminal|1289_endp
;
f 801
SoftPBX: alias 801 not found!
f h323:Wei
RCF|10.1.1.10:1720|800:dialedDigits=Wei:h323_ID|terminal|1289_endp
;
```

- FindVerbose, fv
  Busca el detalle de un endpoint registrado mediante un alias o un prefijo. Para buscar una alias de un tipo específico (h323_ID, dialedDigits), anteponga el tipo de alias (h323, e164, url, email) al alias, seguido por dos puntos (:).

  Formato:

  ```
  FindVerbose Alias
  RCF|IP:Port|Aliases|Terminal_Type|EndpointID
  Registration_Time C(Active_Call/Connected_Call/Total_Call) <r>
  [Prefixes: ##] (gateway only)
  ;
  ```

  Ejemplo:

  ```
  fv 02
  RCF|10.0.1.100:1720|TFN:h323_ID|gateway|4037_CGK1
  Wed, 26 Jun 2002 17:47:29 +0800 C(0/84/120) <1>
  Prefixes: 02,09
  ;
  ```

- UnregisterIP
  Obliga a desregistrar un endpoint mediante su dirección IP y su puerto de señalización de llamada.

  Formato:

  ```
  UnregisterIP IP[:Port]
  ```

  Ejemplo:

  ```
  UnregisterIP 10.0.1.31:1720
  URQ|10.0.1.31:1032|1326_endp|maintenance;
  SoftPBX: Endpoint 10.0.1.31:1720 unregistered!
  ```

- UnregisterAlias
  Obliga a desregistrar un endpoint mediante uno de sus alias. Para emparejar un alias de un tipo específico (h323_ID, dialedDigits), anteponga el nombre del tipo de alias (h323, e164, url, email) al alias, seguido por dos puntos (:).

  Formato:

  ```
  UnregisterAlias Alias
  ```

  Ejemplo:

  ```
  UnregisterAlias 601
  URQ|10.0.1.31:1032|1326_endp|maintenance;
  SoftPBX: Endpoint 601 unregistered!
  ```

- UnregisterAllEndpoints
  Obliga a desregistrar todos los endpoints registrados.

  Formato:

  Ejemplo:

  ```
  UnregisterAllEndpoints
  URQ|10.0.1.7:1024|1325_endp|maintenance;
  URQ|10.0.1.8:1024|1322_endp|maintenance;
  URQ|10.0.1.32:1032|1324_endp|maintenance;
  ```

```
URQ|10.0.1.36:1032|1323_endp|maintenance;
URQ|10.0.1.42:1032|1318_endp|maintenance;
Done
;
```

- DisconnectCall
  Desconecta una llamada mediante su número.(interno, número de llamada asignada por el gatekeeper, no el número de teléfono del emisor ni del receptor).

  Formato:

  ```
  DisconnectCall Number
  ```

  Ejemplo:

  ```
  DisconnectCall 1533
  ```

- DisconnectIP
  Desconecta todas las llamadas de un endpoint mediante su IP y su puerto de señalización de llamada.

  Formato:

  ```
  DisconnectIP IP[:Port]
  ```

  Ejemplo:

  ```
  DisconnectIP 10.0.1.31:1720
  ```

- DisconnectAlias
  Desconecta todas las llamadas de un endpoint mediante uno de sus alias. Para emparejar un alias de un tipo específico (h323_ID, dialedDigits), anteponga el nombre del tipo de alias (h323, e164, url, email) al alias, seguido de dos puntos (:).

  Formato:

  ```
  DisconnectAlias Alias
  ```

  Ejemplo:

  ```
  DisconnectAlias 601
  ```

- ClearCalls
  Desconecta todas las llamadas existentes en el gatekeeper.

- GK
  Muestra información del gatekeeper padre (parent gatekeeper).

- Trace
  Establece el nivel de rastreo de salida para la interfaz de estado. Éste controla qué mensajes son enviados a este cliente:

  - trace 0 o trace min
    Solamente respuestas directas a comandos y notificaciones de recarga (reload).
  - trace 1 CDRs, respuestas directas a comandos y notificaciones de recarga (reload).
  - trace 2 or trace max
    Muestra todo (RAS, CDRs, respuestas directas a comandos y notificaciones de recarga (reload), etc).

- Debug
  Solamente utilizado para propósito de depuración. Opciones:
  - trc [+|-|n]
    Muestra/modifica el nivel de rastreo.
  - cfg SEC PAR
    Lee y muestra un parámetro de configuración de una sección.
  - set SEC PAR VAL
    Escribe un valor a un parámetro de configuración de una sección.
  - remove SEC PAR
    Remueve el valor de un parámetro de configuración de una sección.
  - remove SEC
    Remueve una sección.
  - printrm VERBOSE
    Muestra todos los registros de un endpoint removido.

  Ejemplos:

  ```
  debug trc 3
  debug set RoutedMode H245Routed 1
  ```

- Who
  Muestra todos las personas que se encuentran en el puerto de estado.

- RouteReject
  Termina esta llamada sobre una cola virtual. Este comando es utilizado como respuesta a un evento RouteRequest (ver más adelante). El CallingEndpointID y el CallRef deben ser devueltos tal como están en el correspondiente RouteRequest. El parámetro CallID es opcional; si se dá, éste tiene que tener el mismo formato que el señalado por el RouteRequest con el parámetro SignalCallID=1.

  Formato:

  ```
  RouteReject CallingEndpointID CallRef [CallID]
  ```

  Ejemplo:

  ```
  RouteReject endp_4711 1234
  ```

- RouteToAlias, rta
  Enruta esta llamada sobre una cola virtual hacia el alias especificado. Este comando es utilizado como respuesta a un evento RouteRequest (ver más adelante). El CallingEndpointID y el CallRef deben ser devueltos tal como están en el correspondiente RouteRequest. El parámetro CallID es opcional; si se dá, éste tiene que tener el mismo formato que el señalado por el RouteRequest con el parámetro SignalCallID=1.

  Formato:

  ```
  RouteToAlias Alias CallingEndpointID CallRef [CallID]
  ```

  Ejemplo:

  ```
  RouteToAlias Suzi endp_4711 1234
  ```

- RouteToGateway, rtg
  Enruta esta llamada sobre una cola virtual hacia un alias especificado y establece el destinationSignallAddress. Este comando es utilizado como una respuesta al evento RouteRequest (ver más adelante). Usted puede utilizar este comando para rutear llamadas hacia gateways fuera de la zona (out-of-zone) o MCUs no registrados con el gatekeeper. Asegûese que las políticas 'vqueue' y 'explicit' estén en efecto para esas llamadas. El CallingEndpointID y el CallRef deben ser devueltos tal como están en el correspondiente RouteRequest. El parámetro CallID es opcional; si se dá, éste tiene que tener el mismo formato que el señalado por el RouteRequest con el parámetro SignalCallID=1.

  Formato:

  ```
  RouteToGateway Alias IP:Port CallingEndpointID CallRef [CallID]
  ```

  Ejemplo:

  ```
  RouteToGateway Suzi 192.168.0.50 endp_4711 1234
  ```

- Exit, q
  Cierra el puerto de estado.

- TransferCall
  Trasfiere una llamada establecida desde el alias A hacia el alias B. En un momento dado el alias A está hablando con el alias X, entonces el alias A está hablando con el alias B después del TransferCall.

  Actualmente éste trabaja solamente con endpoints que soportan mensajes Q.931 Facility (por consiguiente éste comando no funciona con Netmeeting).

  Formato:

  ```
  TransferCall Source-Alias New-Destination-Alias
  ```

  Ejemplo:

  ```
  TransferCall Frank Peter
  ```

13.3. Mensajes de Referencia

Esta sección describe los mensajes de salida hacia la interfaz de estado.

- GCF|IP|Aliases|Endpoint_Type;
  El gatekeeper recibe un GatekeeperRequest (GRQ) y responde con un GatekeeperConfirm (GCF).

- GRJ|IP|Aliases|Endpoint_Type|RejectReason;
  El gatekeeper recibe un GatekeeperRequest (GRQ) y responde con un GatekeeperReject (GRJ).

- RCF|IP:Port|Aliases|Endpoint_Type|EndpointID;
  El gatekeeper recibe un RegistrationRequest (RRQ) y responde con un RegistrationConfirm (RCF).

- RRJ|IP|Aliases|Endpoint_Type|RejectReason;
  El gatekeeper recibe un RegistrationRequest (RRQ) y responde con a RegistrationReject (RRJ).

- ACF|Caller_IP:Port|Caller_EndpointID|CRV|DestinationInfo|SrcInfo|IsAnswered[|CallID];
  El gatekeeper recibe un AdmissionRequest (ARQ) y responde con un AdmissionConfirm (ACF). El CallID es solamente enviado cuando el SignalCallId=1 está configurado.
- ARJ|Caller_IP:Port|DestinationInfo|SrcInfo|IsAnswered|RejectReason[|CallID];
  El gatekeeper recibe un AdmissionRequest (ARQ) y responde con un AdmissionReject (ARJ). El CallID es solamente enviado cuando el SignalCallId=1 está configurado.
- DCF|IP|EndpointID|CRV|DisengageReason[|CallID];
  El gatekeeper recibe un DisengageRequest (DRQ) y responde con un DisengageConfirm (DCF). El CallID es solamente enviado cuando el SignalCallId=1 está configurado.
- DRJ|IP|EndpointID|CRV|RejectReason[|CallID];
  El gatekeeper recibe un DisengageRequest (DRQ) y responde con un DisengageReject (DRJ). El CallID es solamente enviado cuando el SignalCallId=1 está configurado.
- LCF|IP|EndpointID|DestinationInfo|SrcInfo;
  El gatekeeper recibe un LocationRequest (LRQ) y responde con un LocationConfirm (LCF).
- LRJ|IP|DestinationInfo|SrcInfo|RejectReason;
  El gatekeeper recibe un LocationRequest (LRQ) y responde con un LocationReject (LRJ).
- BCF|IP|EndpointID|Bandwidth;
  El gatekeeper recibe un BandwidthRequest (BRQ) y responde con un BandwidthConfirm (BCF).
- BRJ|IP|EndpointID|Bandwidth|RejectReason;
  El gatekeeper recibe un BandwidthRequest (BRQ) y responde con un BandwidthReject (BRJ).
- UCF|IP|EndpointID;
  El gatekeeper recibe un UnregistrationRequest (URQ) y responde con un UnregistrationConfirm (UCF).
- URJ|IP|EndpointID|RejectReason;
  El gatekeeper recibe un UnregistrationRequest (URQ) y responde con un UnregistrationReject (URJ).
- IRQ|IP:Port|EndpointID;
  El gatekeeper envía un InfoRequest (IRQ) hacia un endpoint para consultarle si éste está aún en actividad. El endpoint debe responder con un InfoRequestResponse (IRR) inmediatamente.
- URQ|IP:Port|EndpointID|Reason;
  El gatekeeper envía un UnregistrationRequest (URQ) hacia un endpoint para cancelar su registro. El endpoint debe responder con un UnregistrationConfirm (UCF).
- CDR|CallNo|CallId|Duration|Starttime|Endtime|CallerIP|CallerEndId| \
  CalledIP|CalledEndId|DestinationInfo|SrcInfo|GatekeeperID; Después de una llamada desconectada, el registro de detalle de llamada (call detail record) es mostrado (en una línea).
- RouteRequest|CallerIP:Port|CallerEndpointId|CallRef|VirtualQueue|CallerAlias[|CallID];
  Petición para que una aplicación externa enrute una llamada entrante sobre una cola

virtual. Esto puede ser hecho con los comandos RouteToAlias o RouteReject. El CallID es solamente enviado cuando el SignalCallId=1 está configurado.

13.4. Filtrado del Puerto de Estado

Esta sección facilita el control de la cantidad y tipo de los mensajes de salida presentados al usuario final. El Filtrado (Filtering) se realiza utilizando expresiones regulares, las mismas que son utilizadas para decidir si incluír (mostrar) o excluír (ignorar) un mensaje de salida. El control del Filtrado (Filtering) es realizado utilizando el siguiente conjunto de comandos:

- addincludefilter REGEX
  Agrega una expresión regular a la lista de lo que se va a incluír (include list)
- addexcludefilter REGEX
  Agrega una expresión regular a la lista de lo que se va a excluír (exclude list)
- removeincludefilter INDEX
  Quita el filtro dado INDEX de la lista de inclusiones (include list)
- removeexcludefilter INDEX
  Quita el filtro dado INDEX de la lista de exclusiones (exclude list)
- filter 1|0
  Habilita/Deshabilita el filtrado de Mensajes
- printincludefilters
  Presenta la lista de filtros incluídos (include filter list)
- printexcludefilters
  Presenta la lista de filtros excluídos (exclude filter list)

Para habilitar el uso de filtros predeterminados, se ha añadido la sección 4.2.3 ([GkStatus::Filtering]). En esta sección, los usuarios pueden poner sus filtros predeterminados que serán cargados cuando se inicie el Puerto de estado.

Ejemplo:

```
[GkStatus::Filtering]
IncludeFilter=.+
ExcludeFilter=.RQ
```

Cuando el Filtrado es activado, mediante el comando filter 1, se mostrarán todos los mensajes, pero no aquellas líneas con ARQ, LRQ etc. El mismo efecto puede lograrse utilizando el comando line:

```
addincludefilter .+
addexcludefilter .RQ
filter 1
```

Tenga en cuenta que habilitar el Filtrado cuando no se han definido filtros, automáticamente excluye todos los mensajes de salida.